A Survey on Unicast Congestion Control Protocols for Media Traffic

Mohammad A. Talaat, Magdi A. Koutb, Hoda S. Sorour

Abstract—Congestion control for streamed media traffic over Internet is a challenge due to the sensitivity of such traffic towards oscillations in the rate of streaming. This challenge motivated researchers over the last decade to develop a number of congestion control protocols that suit the media traffic and provides TCP-friendliness for both unicast and multicast communications. This paper presents a discussion for the congestion control protocols categorization characteristics, elaborates the TCP-friendliness concept then a state-of-the-art for the unicast congestion control protocols designed for media traffic is presented. The paper points the pros and cons for each of these protocols, and evaluates their algorithms characteristics.

Index Terms—Congestion Control, Unicast, Media Traffic

I. INTRODUCTION

CONGESTION control over Internet, for both of unicast and multicast media traffic, has been an active area of research in the last decade as seen in [1]. This is due to the booming increase in the audiovisual traffic during the newly born era of digital convergence There exists a variety of Internet applications nowadays built on its capability of streaming media either in real-time or on demand such as video streaming and conferencing, voice over IP (VoIP), and video on demand (VoD). The number of users for these applications is constantly growing.

Unicast is a one-to-one form of communication over IP networks meanwhile multicast is one-to-many. Mulicast is advantageous over unicast especially in bandwidth saving, but unicast is still the widely more spread communication form over Internet so far.

Media is usually a payload for UDP packets, rather than TCP, to benefit from the simplicity and robustness of this protocol. Other types of packets such as HTTP, FTP, and SMTP use TCP as their transport-layer protocol. The coexistence of TCP and non-TCP flows on the same link causes a problem of unfairness in the link bandwidth distribution in cases of congestion. This problem arises when non-TCP flows fail to act in a *TCP-friendly* way and consequently occupies more network resources than their fair share.

Congestion status leads routers to drop the packets at the end of their queues through an unwanted *drop-tail* action. Hence, TCP attempts to avoid congestion via adapting its sending rate according to an additive increase/multiplicative decrease algorithm (AIMD). TCP flows tend to decrease their sending rate when congestion is noticed to one half for example, according to AIMD. Non-TCP flows concurrently may increase their sending rates to occupy the bandwidth released by TCP flows, thus, retaining the same status of congestion.

This unfair situation will lead to an Internet *collapse*, where the available bandwidth will be filled with packets that are discarded, due to congestion, before reaching their destinations. Therefore, several *TCP-friendly* protocols were developed for non-TCP traffic to avoid Internet *collapse* and relieve congestion at the same time. These protocols tend to work in a compatible way with the AIMD mechanism used by TCP to face congestion, and consequently avoid any TCP starvation.

Efforts made by researchers to develop *TCP-friendly* protocols targeted both unicast and multicast communications. Protocols developed for muticast traffic were more complex than that of unicast. It was observed lately that *TCP-friendly* unicast protocols gained more of the researchers' attention than multicast ones due to their wider spread of usage over Internet. Congestion control over high speed networks was also a target of research as shown in [2]

This paper presents the classification features for the congestion control protocols developed and tested, so far, by researchers. Then the paper surveys the unicast category of these protocols, and namely those that suit the media traffic transfer. This survey is followed by an evaluation for the state-of-the-art in this domain, and to encourage further research on *TCP-friendliness*.

The rest of this paper is organized as follows: Section II elaborates the TCP-friendliness concept, and the TCP AIMD mechanism as well as its throughput model. Section III discusses the different parameters of classifying the

M. A. Talaat is with the Information Technology Institute, Cairo, Egypt (e-mail: moadly@mcit.gov.eg).
M. A. Koutb, Prof., is with the Faculty of Electronic Engineering, Menouf, Egypt.
H. S. Sorour is with the Faculty of Electronic Engineering, Menouf, Egypt,

Manuscript received March 5, 2009
Manuscript revised March 20, 2009

congestion control protocols for media traffic. Section IV describes briefly a number of the unicast category of protocols and their mechanisms of work. Section V evaluates these protocols in terms of the algorithm adopted by each of them, and finally section IV concludes.

II. TCP-FRIENDLINESS

A. *TCP and AIMD*

TCP is a transport-layer, reliable and connection-oriented unicast protocol. TCP's advantage of reliability over UDP lies in the error control property, it also applies flow and congestion control via the AIMD mechanism. TCP keeps a congestion window *cwnd* that holds all the packets to be sent at the beginning of a session, it starts then to send some packets from the window consequently and waits for their acknowledgment *ACK* to arrive. When receiving the acknowledgement of reception for those packets, the *cwnd* shifts to replace them by other ones not sent yet. TCP integrates a *slow-start* that doubles the rate of sending every round-trip time (RTT) to fully and fairly occupy the available bandwidth.

After reaching the steady state TCP moves to AIMD to detect any additional free bandwidth and to react to congestion. After each RTT, TCP either increases its cwnd by one when no packet loss is notified, or decreases it by one when a packet loss is notified. AIMD halves its sending cwnd when three duplicate acknowledgments arrive indicating an event of congestion as shown in equation (1) [3].

I: $w_{t+RTT} = w_t + \alpha;\ \alpha >$

D: $w_{t+\delta t} = (1-\beta)w_t;\ 0 < \beta <$

(1)

B. *TCP Throughput Model*

TCP throughput *T* is dependent on the following parameters: *RTT*, retransmission time-out value *RTO*, segment size *S*, and the rate of packet loss *P*. Equation (2) [4] gives a derived estimate of TCP throughput *T* in the steady state. This equation is the complex model of TCP throughput.

$$T = \min\left(\frac{W_m S}{RTT}, \frac{S}{RTT\sqrt{\frac{2bP}{3}} + RTO\min\left(1, 3\sqrt{\frac{3bp}{8}}\right)p(1+32p^2}\right.$$

(2)

Where is the number of packets acknowledged by each ACK and W is the maximum window size of *cwnd.*

C. *TCP-Friendliness*

TCP-friendliness is measured through the effect of a non-TCP flow on the competing TCP flows under the same conditions regarding throughput and other parameters. A non-TCP unicast flow is said to be *TCP-friendly* if it does not affect the long term throughput for any of the co-existing TCP flows by a factor that is more than that done by a TCP flow under the same conditions. A multicast flow is said to be *TCP-friendly* if for each sender-receiver pair of the multicast flow is seen to be seperately *TCP-friendly*.

Note that the above definitions ensure that TCP flows are not treated unfairly by non-TCP flows sharing the same congested link. However, it is not guaranteed that all the TCP and non-TCP flows are running with the same throughput on the bottleneck link holding them. It is probable also that the TCP flows themselves vary in their throughput depending on the RTT of each of them.

III. CLASSIFICATION OF CONGESTION CONTROL PROTOCOLS

Congestion control protocols are classified into many categories according to a number of features in their mechanism of work. The following lines show the valid categories of classification.

A. *Window-Based versus Rate-Based*

TCP-friendly protocol can adapt either its congestion window *cwnd* size or its transmission rate in accordance with congestion incidents. Both adaptations lead to change in the offered network load of the protocol, hence protocols can be divided into window based protocols and rate based protocols.

Window-Based protocols are built upon the congestion window-based mechanism, and use the congestion window *cwnd* at the sender or receiver side. A slot in that window is reserved for each packet; this slot is free only when the sent packet is acknowledged to be received, and transmission is allowed only when free slots are valid. The size of *cwnd* increases in absence of congestion and decreases when congestion occurs. A suggestion for adjusting the window-based congestion control in run-time is proposed in [5]

Rate-Based protocols are built upon adapting their rate of transmission according to some integrated feedback algorithm that notifies about congestion when existent. Rate-based algorithms can be subdivided into simple AIMD mechanisms and model-based congestion control. Simple AIMD schemes results in a saw-tooth throughput shape, thus, this type of schemes usually is not fully compatible with the streaming media applications. Model-based algorithms use a TCP model to control congestion instead of the TCP-like AIMD. Model-based protocols aim to smooth the sending rate of the protocols using it, in the short-term, to be more suitable for the media traffic type, this makes model-based category TCP-friendly over long-term scale. Model-based congestion control may not resemble the mechanism used by TCP.

Current researches tend to make the adjustment rate mechanisms ensure the fairest competition between TCP and non-TCP flows.

B. Unicast versus Multicast

Both unicast and multicast traffic protocols need to be *TCP-friendly*, but obviously the design of the multicast TCP-friendly scheme is more complex than that of the unicast. This complexity lies in the *heterogeneity* problem that is due to the variation in network conditions at one of the receiver's side in a way that differs from that at another receiver's side.

This variation becomes problematic as the number of receivers gets larger. The sender in this case should react to the congested network state for each one of the receivers by decreasing the sending rate in the whole multicast session (due to the arrival of a congestion notification from only one of the receivers). This decrease may not be approved by the rest of the receivers that suffer no congestion at this moment, and do not like to have any quality degradation as a result for this decrease in rate. Moreover, responding to every congestion incident at a specific receiver's side leads to fluctuations in the sending rate that is most undesirable for media traffic. Responding to the overall average congestion rate at all the receivers' sides is not a much better solution as well, since fluctuations in the sending rate will occur also when applying this option, a survey on the multicast congestion control protocols can be found in [6].

C. Single-rate versus Multi-rate

Single-rate is typically the mechanism adopted by all the unicast congestion control protocols. Transmission in unicast has only one recipient, so it adapts its sending rate according to this recipient's status. Multicast transmission can adopt the single-rate approach as well, where the sender streams data with the same rate to all recipients of the muticast group. This rate is chosen to suite the bottleneck of one of the receivers, but meanwhile it may not be the right one for all of the other recipients that suffers less congestion. Hence, obviously, this approach limits the scalability property of the mulilticast protocol towards the network conditions of each recipient separately.

Multi-rate transmission allows for more scalability for the multicast communication, where each of the multicast session routes can be assigned a specific rate of transmission according to its existing network conditions. This scalability is more appropriate for large set of multicast recipients that suffer from increased *heterogeneity* commonly found over Internet. Multi-rate congestion control uses the layered multicast approach, since multi-layering enables the sender to divide data into different layers to be sent to different multicast groups. Every receiver has to join the largest possible number of groups permitted by the bottleneck in the way to sender. The quality of data sent to this receiver is higher when joining more groups. This feature is most obvious in multicast video sessions where the more groups the recipient subscribes in, the more layers the recipient receives, and the better the quality of video is. Meanwhile, for other bulk data, additional layers decrease the transfer time. Byu sing this mechanism, congestion control is achieved implicitly through the group management and routing mechanisms of the underlying multicast protocol.

D. End-to-end versus Router-supported

Internet is known to be the best-effort IP network that provides no congestion control. Most of the congestion control algorithms are designed to work on the end-to-end basis that needs no support from the network to realize TCP-friendliness. End-to-end congestion control schemes have the advantage of being ready for direct implementation on today's Internet. This category can be subdivided into sender-driven and receiver-driven schemes. In sender-driven schemes, the sender tunes its window size according to the information reaching its side about the network congestion in order to reach TCP-friendliness. Receivers can only send feedback but the rate tuning decision is totally seen as the sender's responsibility. Receiver-driven congestion control can be seen clearly in the layered approach. It is the receiver's decision to subscribe or unsubscribe from extra layers based on the network congestion status.

The mission of congestion control, especially the part of fairly sharing the network resources, can be facilitated by implementing part of its mechanisms over the network devices and not only at its terminals. Schemes that rely on an embedded functionality in the connecting routers are called router-supported. Multicast protocols in particular depends on some of the network information regarding round-trip times and management of groups of receivers. Modification of the routers' queuing mechanism can greatly help the design and implementation of a multicast congestion control scheme.

End-to end schemes has the disadvantage of being dependable on the systems implemented on Internet terminals. A disadvantage in end-to-end control is that it allows greedy users to acquire all the available bandwidth via applications using non-TCP-friendly mechanisms, and in the absence of any router control.

Router-supported mechanisms are to drop the non-TCP-friendly packets with a probability higher than that of dropping the TCP-friendly packets. Router-supporting has the disadvantage of being costly to deploy over the widely spread infrastructure of today's Internet in terms of money, time, and effort.

IV. Unicast Congestion Control Protocols

This section is a state-of-the-art survey for the unicast congestion control protocols.

A. *RAP*

The Rate Adaptation Protocol (RAP) as described in [7] adopts a simple AIMD mechanism to adapt unicast flows. An acknowledgment is sent by each data packet that is used to detect a packet loss and calculate the RTT. When RAP senses congestion, it decreases the sending rate to its half. If no congestion is there, the rate is increased by one packet every RTT. This attitude resembles the AIMD used by TCP, every RTT a decision is taken either by increase of rate or decrease. To smooth the sending rate RAP uses the ratio between the short-term RTT average and long-term RTT average to reach additional fine-grained delay-based congestion avoidance. This ratio is the base of modifying the inter-packet gap IPG between data packets. Fine-grained rate adjustment results in a smooth change in the rate of sending.

RAP's decision of halving the sending rate resembles that taken by TCP when three duplicate acknowledgments arrive, thus the rate changes in RAP looks like that of TCP sending rate that suffers few timeouts. Timeouts are not taken into account in RAP; this makes RAP act aggressively when timeouts dominate the TCP throughput.

B. *LDA+*

The Loss-delay Based Adaptation Algorithm (LDA+) described in [8] relies on the feedback report sent by the Real-time Transport Control Protocol RTCP working concurrently with Real-time Transport Protocol RTP. This report supplies the sender with information about the receiver's arriving traffic status; hence LDA+ has no specific mechanism of notification to implement other than using this report. LDA+ is an AIMD algorithm that uses its own parameters in calculating the factor by which the rate is increased or decreased. This factor is calculated dynamically according to the concurrent network conditions. LDA+ calculates an estimate as well for the bottleneck bandwidth based on the time interval between the receipts of two packets that were sent back-to-back. The factor of additive increase in LDA+ ensures the following three constraints: i) Flows running at a low bandwidth are more probable to increase their rate than flows running at higher bandwidth. ii) Estimated bottleneck is not surpassed. iii) LDA+ flows do not increase their bandwidth faster than TCP flows.

When a loss is reported, the sending rate is decreased by multiplying by a factor equals $1 -$, where is the loss rate. The rate can be reduced to the most, which is the rate used by the TCP model. LDA+ may use the maximum of the AIMD rate and the model based rate of equation (2) leading to be more aggressive than TCP.

Simulations showed that LDA+ achieves fairness to TCP LDA+ design was eased due to its dependence on the existing reporting mechanism of RTCP. LDA+ is advantageous in keeping its rate below the estimated bottleneck. LDA+ disadvantage lies in the RTCP reports that are generated frequently but separated by seconds; this makes the dependent LDA+ mechanism somehow slow in reacting to congestion. RTCP reports is limited in its lower value of loss that can be recorded in it, hence LDA+ can be mislead by a zero loss value in this report and consequently increase its sending rate, meanwhile the network is suffering from low loss values that were not sensed by RTCP. This limitation can lead LDA+ to occupy more than a fair share.

C. *TFRCP*

TCP-Friendly Rate Control Protocol TFRCP is model based that uses the complex model of equation (2) to adjust the rate of sending in a TCP-friendly manner as described in [9]. The model parameters in TFRCP are recalculated every fixed time round aiming to reach TCP-friendliness. If no packet loss is detected during a round, the sending rate is doubled the next round. Otherwise the sending rate follows the TCP model. TFRCP mandates that each data packet is acknowledged by the receiver. The recalculation duration is a base in this protocol, and since the duration of recalculation is fixed, the protocol is not flexible enough to adapt to the network variations. Doubling the rate of sending due to loss absence is an aggressive decision if compared to the TCP action.

TFRCP does not suffer only from being unfair to TCP, but it also suffers from the rapid fluctuations in the rate of sending. This happens when some loss is detected where the rate follows the TCP model to be below the bottleneck bandwidth leading to absence of loss or trivial losses, which in turn causes the doubling of the sending rate. This doubling leads to packet loss existence that causes consequently to dropping the rate to follow the equation and so on.

D. TFRC

The TCP-friendly rate control protocol TFRC [10] is an enhanced version of TFRCP. TFRC uses the complex TCP rate adaptation equation like TFRCP; meanwhile TFRC adopts a simpler method for gathering its required parameters. TFRC was originally designed to serve unicast communications, but it can also be modified to support multicasting. TFRC demanded a loss rate estimator that can be most expressing, so it used the Average Loss Interval method to calculate this loss estimator. This method was the best to fulfill the simplicity requirement in TFRC. The loss rate is estimated based on the loss intervals, covering the number of packets between consecutive loss events. The average of loss intervals over a specific number of loss events is calculated, decaying weight method is used so that the old loss events are less in their effect when calculating this average. The loss rate is calculated as the inverse of the average loss interval value.

TFRC design ensured that no single loss event can strongly affect the loss rate, and that the loss rate is fast in response to long intervals that is free from losses. RTT is measured in TFRC in the standard way by reading the timestamps of packets across the network and sending them to sender.

TFRC sender passes through the slow start period similar to that of TCP until reaching its maximum available share of the bandwidth without losses. Slow start is ended by a loss event. Every RTT, the receiver changes its parameters to their new values and sends a state report to the sender. The sender computes its new fair rate and tunes its sending rate accordingly. TFRC can also use the delay-based congestion control model in the environments that do not support the complex TCP equation; this is done via tuning the IPG. An attempt was made in [11] to make TFRC run using linear throughput equation. TFRC managed to achieve a relatively stable sending rate while keeping sufficient responsiveness to the co-existing traffic.

E. Binomial Algorithms

Design of the binomial algorithms [3] was motivated by the suffering of the audio and video streaming applications from the drastic reduction of the transmission rate upon each loss incident. The authors presented a class of non-linear congestion control algorithms that generalize the TCP-style AIMD. They made the increase inversely proportional to a power of k of the current window (for TCP $k = 0$), also they made the decrease proportional to a power l of the current window (for TCP $l = 1$). They showed that an infinite number of algorithms can exist satisfying the condition that $k + l = 1$, and that all of them can converge to fairness provided k, $l > 0$. Authors focused on two algorithms which are Inverse Increase /Additive Decrease IIAD ($k = 1$, $l = 0$) and SQRT ($k = l = 0.5$). These algorithms are called binomial due to their dependence on two different algebraic terms with different exponents. Binomial algorithms are simple; moreover, reducing the sending rate in case of decrease using the value of $l < 1$ is generally less drastic than AIMD. This feature makes binomial algorithms more appropriate for the audio and video streaming applications over Internet, especially for k, l not equal to 0, 1 respectively.

Simulations showed that binomial algorithms interact well with TCP across Random Early Detection RED gateways. TCP-compatible binomial algorithms such as IIAD and SQRT achieve a higher long-term throughput than TCP over drop-tail bottleneck gateways, because of higher average buffer occupancy. Authors recommended the elimination of the drop-tail gateways from the infrastructure of Internet for many reasons, among which was the promising behavior seen from the active queue management algorithms like RED, especially on the interaction between TCP and the binomial algorithms.

F. SMCC

The Streaming Media Congestion Control Protocol SMCC presented in [12] works on the basis of bandwidth estimation concept. The rate of packets transmission in a certain connection is adaptively adjusted according to the dynamic share of bandwidth of a connection. Bandwidth estimation is made using algorithms similar to that of TCP Westwood [13]. SMCC does not pass by the slow start phase of TCP, hence it avoids the oscillations in the rate of transmission experienced by TCP, and consequently SMCC is suitable for streaming media applications as a congestion control protocol.

SMCC does not use congestion window, meanwhile it adopts the linear bandwidth probing of TCP through adjusting its sending rate to reach congestion avoidance. One extra packet per RTT is sent in SMCC as long as no congestion is detected. When congestion is encountered, the bandwidth reduces to the current Bandwidth Share Estimate BSE, and linear probing continues. The TCP algorithm of dropping the sending rate by one packet per RTT, when a timeout happens, is never followed unless the BSE imposes it.

BSE in SMCC is the estimate of the rate to be used by the sender in order to share the bandwidth fairly among other flows. SMCC exponentially averages the rate samples to calculate the BSE in the same manner used by TCP Westwood. SMCC utilizes the inter-arrival time between two consecutive packets at the receiver side to calculate a sample rate on the forward connection path. TCP Westwood uses the arrivals of acknowledgments to the sender for the same calculation; this makes SMCC advantageous in measuring the rate on the forward path while neglecting the effect of the reverse path congestion.

SMCC is fair so that a number of SMCC flows can share the available bandwidth equally; moreover SMCC is friendly to TCP New Reno protocol. SMCC is also robust in responding to packet losses that is due to random errors which is characteristic in wireless connections; this makes SMCC advantageous due to the growing Internet wireless access

G. MTFRCC

Media and TCP-friendly Rate-based Congestion Control as proposed in [14] is mainly designed for scalable video streaming over Internet. MTFRCC integrates the two new following mechanisms: i) utility-based model using the rate-distortion function as the application utility measure for optimizing the over-all video quality; and ii) two-timescale approach of rate averages (short-term and long-term) to satisfy both media and TCP-friendliness. The distortion function expresses the quality of the rendered scalable video stream, while the utility function varies according to the video coding algorithm used. MTFRCC design is based on the gradient projection algorithm. The packet loss rate is the congestion information utilized by MTFRCC. The performance of MTFRCC was evaluated by authors in terms of the following metrics: Fairness to TCP, responsiveness, aggressiveness, smoothness, and overall video quality.

Simulations and tests showed that MTFRCC achieved smoother rates than TFRC. It was also found that MTFRCC is competing with TFRC in terms of responsiveness and aggressiveness when facing sudden changes in the available bandwidth, MTFRCC resulted in fewer oscillations in the sending rate during transitional periods. MTFRCC achieved a better overall video quality than that of TFRC in different levels of congestion and with less variance compared to TFRC. MTFRCC proved as well that introducing the property of media-awareness to the congestion control algorithm leads to better quality for the streamed video.

H. DMSCC

Distributed Media Streaming Congestion Control DMSCC as described in [15] aims to control congestion for systems with multiple senders that collaboratively and simultaneously stream media content to a receiver. Each of these senders generates a separate flow to the receiver that is TCP-friendly. Greedy Users may increase the number of TCP-friendly flows to acquire a larger share of the bottleneck bandwidth, this behavior degrades the over-all network performance, and opposes new challenges to congestion control.

DMSCC introduces the task-level congestion control concept. It enforces a group of flows belonging to the same task to be TCP-friendly instead of achieving friendliness on each flow individually. DMSCC observes congestion in a distributed media streaming system, identifies the set of flows causing it, and then dynamically tunes these flows to make their over-all throughput TCP-friendly. To achieve this goal, DMSCC had to reduce the throughput for a given flow by a factor of β using AIMD, where $0<\beta<1$

DMSCC is a receiver-driven congestion control protocol, where receivers pull data from sender by sending requests having different sequence numbers as an identifier. Each of the various connections at the receiver side is controlled by an AIMD loop, like that of TCP, the increasing factor of this loop is controlled by the DMSCC module of this receiver. DMSCC assumed that flows on the same bottleneck links suffer the same loss-rate.

Simulations showed that DMSCC was successful in controlling congestion on the task level as proposed, and could achieve an overall TCP-friendly throughput for a number of flows for a distributed media streaming system. DMSCC performance was limited in the cases of bursty and frequent packet losses.

I. SSVP

Scalable Streaming Video Protocol SSVP proposed in [16] is an end-to-end protocol working on top of UDP and optimized for unicast video streaming in the real-time. SSVP adopts the AIMD mechanism by tuning the sending rate via controlling IPG. SSVP achieved TCP-friendliness and moreover, maintained the smoothness of AIMD oscillations in throughput through its technique in adjusting the AIMD parameters and the IPG control.

SSVP mechanism enhances the performance of UDP through sending acknowledgments for the datagrams received in the form of control packets that contain no data. Those control packets do not trigger retransmissions to add reliability to UDP, but they are used to determine the RTT and estimate the available bandwidth accordingly. The sender has to adjust the IPG every RTT through calculating the ratio of the number of control packets with congestion indicator to the total number of control packets. If this ratio is larger than a specific threshold, the sender infers congestion and reduces its transmission rate via increasing the IPG.

Experimental results showed that SSVP adapts to the network vagaries and enhances remarkably video streaming in real-time. SSVP is tested along with a video layered adaptation scheme that uses buffering at the receiver side, and adapts video quality to the long-term network variations in bandwidth. A new layer is sent by this mechanism based on the available bandwidth and the size of the receiver buffer in a certain point of time; this inhibits unwanted layer changes that affect the user perceived quality of video. SSVP showed remarkable advancement in video delivery especially under limited bandwidth conditions.

J. TFWC

The TCP-friendly window based Congestion Control mechanism TFWC proposed in [17] is concerned with real-time multimedia streaming applications. Fairness is the main point that TFWC targets through introducing a TCP-like acknowledgment mechanism, while maintaining the TCP throughput equation of calculating the sending rate. TFWC follows the same TCP throughput equation (2) used by TFRC, but whereas TFRC uses this equation in calculating the sending rate as a rate-based algorithm, TFWC uses the same equation to calculate the congestion window size *cwnd* as a TCP-like ack-clocked window-based algorithm. TFWC does not retransmit lost packets as recommended by streaming media applications. The window size can be derived from the TCP throughput equation to get the number of packets   for the new *cwnd* given   as the loss probability (loss rate event).

$$W = \frac{1}{\sqrt{\frac{2p}{3}}+\left(12\sqrt{\frac{3p}{8}}\right)p(1+32p^2)}$$

(3)

TFWC follows the same behavior of TCP until the first packet loss incident takes place. TFWC then starts to calculate the average loss interval to get the loss event rate upon receipt of each acknowledgment. Substituting this rate in equation (3) TFWC computes the new *cwnd* size. Every time an acknowledgement arrives to the sender, the RTT and RTO values are updated. The retransmission timer is set with the calculated timeout value every time a new packet is sent. When the timer expires, the next packet is sent without retransmissions, and then the timer is set with double the latest timeout value.

Simulations showed that TFWC is fairer to TCP than TFRC, especially over congested DSL links that are mostly used by the media streaming applications users.

K. DVRC

The Dynamic Video Rate Control described in [18] is a new streaming protocol that works on top of UDP to adapt video delivery over Internet. DVRC design is based on the user perception sensitivity towards the smoothness and timely playback of the received video frames. DVRC aimed to adjust the rate of video transmission in accordance with the prevailing network conditions. This rate adjustment is made using a number of scale values for the diverse video encoding methods. If no congestion is detected, the scale value can be increased by one, meanwhile, in case of congestion, the scale value is decreased by one. The receiver sends a periodic feedback to the sender through the control packets to report the congestion status of the network. Each generated control packet is updated with the scale value which informs the sender the appropriate transmission rate. DVRC is applicable for the existing video streaming applications.

Experiments showed that DVRC managed to adapt to the network vagaries. DVRC controlled the greedy nature of UDP and maintained friendliness with TCP traffic.

TABLE I
UNICAST CONGESTION CONTROL PROTOCOLS

No	Protocol	End-to-end/ Router Supported	Rate-Based/ Window-Based	Method	Sender-Driven/Receiver-Driven
1	RAP	End-to-end	Rate-Based	AIMD	Sender-Driven
2	LDA+	End-to-end	Rate-Based	AIMD	Sender-Driven
3	TFRCP	End-to-end	Rate-Based	Model-Based	Sender-Driven
4	TFRC	End-to-end	Rate-Based	Model-Based	Sender-Driven
5	Binomial	Router-Supported,	Rate-Based	AIMD	Sender-Driven
6	SMCC	End-to-end	Rate-Based	Model-Based	Sender-Driven
7	MTFRCC	End-to-end	Rate-Based	Model-Based	Sender-Driven
8	DMSCC	End-to-end	Rate-Based	AIMD	Receiver-Driven
9	SSVP	End-to-end	Rate-Based	AIMD	Sender-Driven
10	TFWC	End-to-end	Window-Based	Model-Based	Sender-Driven
11	DVRC	End-to-End	Rate-Based	Model-Based	Receiver-Driven

V. PROTOCOLS EVALUATION

Attempts to achieve TCP-friendliness in the congestion control protocols included other than the above mentioned ones such as the attempt of running the Real-time Transport Protocol (RTP) over the Datagram Congestion Control Protocol (DCCP) [19], as well as an attempt to design a general purpose Stream Control Transmission Protocol (SCTP) [20]

Table I points out the basics of the algorithms adopted by each of the previously discussed protocols. All of the protocols discussed were unicast, end-to-end, rate-based single-rate, and sender-driven protocols with following exceptions: TFWC was the only window based protocol, DMSCC and DVRC were the only receiver-driven protocols, and binomial were the only router-supported protocols. Another suggestion for a user-centric evaluation of such protocols for real-time video transmission is in [21]

VI. Conclusion and Future Work

This paper discussed the congestion control protocol categories and elaborated on the concept of TCP-friendliness and its models. A state-of-the-art for the unicast congestion control protocols for media traffic was presented that covered the protocols developed over the last decade, a characteristic evaluation for the algorithm used by each of these protocols was shown

Our evaluation showed that the rate-based end-to-end congestion control schemes prevails the research in this area. We noticed that the testing of each of the protocols designed after TFRC was done to compare the performance of this protocol namely with TFRC itself. This indicates that TFRC acts as a benchmark for the TCP-friendly protocols when used in media traffic. We can also claim that no protocol has reached the stage of maturity to be the standardized protocol for media streaming.

Our future work will focus on TFRC to enhance its performance in terms of friendliness, smoothness, and fairness via designing, implementing, and testing a novel congestion protocol that can be a step towards reaching the standard congestion control protocol for media traffic over Internet.

References

[1] J. W. Chung, "Congestion control for streaming media," *Ph. D. dissertation*, Polytechnic Inst., Worcester, 2005.

[2] K. Satyanarayan Reddy and Lokanatha C. Reddy, "A survey on congestion control mechanisms in high speed networks," *IJCSNS International Journal of Computer Science and Network Security*, vol. 8, no. 1, 2008, pp. 187 – 195.

[3] D. Bansal and H. Balakrishman "Binomial congestion control algorithms," in *Proc. IEEE INFOCOM'01*, 2001, pp. 631–640.

[4] J. Widmer, R. Denda, M. Mauve, "A survey on tcp-friendly congestion control (extended version)," *Technical Report, Department for Mathematics and Computer Science, University of Mannheim*, Feb. 2001.

[5] G. De Marco, F. Postiglione, M. Longo, "Run-time adjusted congestion control for multimedia: experimental results", *Journal of Interconnection Networks (JOIN)*, vol. 5, no. 3, pp. 249-266, 2004.

[6] A. Matrawy, I. Lambadaris, "A survey of congestion control schemes for multicast video applications," *IEEE Communications Surveys and Tutorials*, 2003.

[7] R. Rejaie, M. Handley, and D. Estrin, "Rap: An end-to-end rate-based congestion control mechanism for real-time streams in the internet," *Proc. IEEE Infocom*, Mar. 1999.

[8] D. Sisalem and A. Wolisz, "Lda+ tcp-friendly adaptation: A measurement and comparison study," *Proc. International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, June 2000.

[9] J. Padhye, D. Kurose, and R. Towsley, "A model based tcp-friendly rate control protocol," *Proc. International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, June 1999.

[10] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proc. ACM SIGCOMM*, Stockholm, Sweden, Aug. 2000, pp. 43 – 56.

[11] Agnieszka Chodorek and Robert R. Chodorek, "Streaming video over tfrc with linear throughput equation," *Poznan University of Technology Academic Journals*, 2007.

[12] N. Aboobaker, D. Chanady, M. Gerla, M. Y. Sanadidi, "Streaming media congestion control using bandwidth estimation," *MMNS 2002*,California,Oct. 2002, pp. 89 – 100 .

[13] M. Gerla, M. Sanadidi, K. Ng, M. Valla, R. Wang, "Efficiency/Friendliness Tradeoff in TCP Westwood," *ISCC 2002*, Taormina, Italy, 2002.

[14] Jinyao Yan, Kostas Katrinis, Martin May, and Bernhard Plattner, "Media- and tcp-friendly congestion control for scalable video streams," *IEEE Transactions on Multimedia*, vol. 8, no. 2, 2006, pp. 196 – 206.

[15] L. Ma, W. Tsang, "Congestion control in distributed media streaming," in *Proc. INFOCOM 2007*, Anchorage, Alaska, USA, May 2007.

[16] Panagiotis Papadimitriou, Vassilis Tsaoussidis, "Ssvp: A congestion control scheme for real-time video streaming," *Computer Networks*, vol. 51, no. 15, 2007, pp.4377 – 4395.

[17] S. Choi and M. Handley, "Fairer tcp-friendly congestion control protocol for multimedia streaming applications," *CoNEXT* 2007, New York, USA, 2007.

[18] P. Papadimitriou and V. Tsaoussidis, "A rate control scheme for adaptive video streaming over internet," *42nd IEEE International Conference on Communications*, Glasgow, Scotland, June 2007.

[19] Colin Perkins and Ladan Gharal, "Rtp and the datagram congestion control protocol," in *Proc. IEEE International Conference on Multimedia and Expo*, Toronto, Canada, July 2006.

[20] Doo-Won Seo, Hyuncheol Kim, "Design of sctp-scf," *IJCSNS International Journal of Computer Science and Network Security*, vol. 8, no. 4, April 2008.

[21] M. Welzl and Werner Stadler, "User-centric evaluation of tcp-friendly congestion control for real-time video transmission," *Elektrotechnik und Informationstechnik*, June 2005.