A CONGESTION CONTROL ALGORITHM FOR THE PLANETARY INTERNET

L. A. Grieco [*] **S. Mascolo** [*]

[*] *Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, Via Orabona, 4 – 70125 BARI, Italy*

Abstract: This paper proposes a new rate based congestion control algorithm that significantly improves the utilization of planetary links with respect to classic TCP congestion control. Ns-2 simulation results have shown that the proposed algorithm provides significant goodput improvements with respect to New Reno, Reno Sack, Reno Fack, Vegas, and Westwood+ TCP in the presence of RTTs larger than 1s and smaller than 2000s and packet loss probability ranging from 0.0001 to 0.01 *Copyright ©2005 IFAC.*

Keywords: Congestion Control, Planetary Internet, Time-delay systems

1. INTRODUCTION

TCP congestion control algorithm has been designed to operate over reliable links with limited range of bandwidths and round trip times (Jacobson, 1988). As a consequence, it does not perform satisfactorily over planetary links (see Fig. 1), which are characterized by very long propagation delays (up to 40min), and extremely high BER (up to 10^{-1}) (Akyildiz *et al.*, 2004).

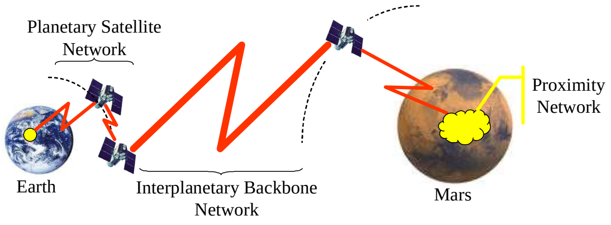

Fig. 1. Example of Interplanetary path.

In fact, TCP congestion control follows an *additive increase* mechanism to grab all available bandwidth and a *multiplicative decrease* mechanism to drastically decrease the window when congestion is revealed by a timeout or reception of 3 duplicate acknowledgments (DUPACKs). The aggressiveness of the TCP probing phase diminishes when the connection Round Trip Time (RTT) increases, moreover TCP interprets packet losses due to unreliable wireless links as symptoms of congestion and consequently shrinks the sending rate also if it is not due (Akyildiz *et al.*, 2004). Westwood TCP and its enhanced variant Westwood+ are a modified version of classic TCP that leave unchanged the probing phase, i.e. the slow-start and congestion avoidance phases, and propose a new mechanism to shrink the congestion window and the slow-start threshold after congestion, which is based on a innovative mechanism to estimate the available bandwidth in an end-to-end fashion (Mascolo *et al.*, 2001; Grieco and Mascolo, 2004). In the paper (Akan *et al.*, 2002) it has been shown that Westwood TCP does not perform satisfactorily over planetary paths, in this paper we will show that also Westwood+ is affected by similar problems. Recently, to overcome the fundamental limitations of TCP over planetary paths, the TCP-Planet congestion control algorithm has been proposed in (Akyildiz *et al.*, 2004). TCP-Planet employs an end-to-end rate-based Additive Increase Multiplicative Decrease (AIMD) congestion control, whose AIMD parameters are tuned by taking into account the connection RTT in order to avoid throughput degradation (Akyildiz *et al.*, 2004). It exploits a stream of low priority packets (Akyildiz *et al.*, 2001) to probe the network available bandwidth and throttles the sending rate by taking into account the ratio $\phi = N_{low}/N_{high}$, where N_{low} (N_{high}) is the number of low (high) priority packets received by the sink at the end of each measurement period. Both N_{low} and N_{high} are sent back to the TCP-Planet sender by using apposite feedback reports. When ϕ is less than a threshold ϕ_d the TCP-Planet sender multiplicatively decreases the sending rate; when $\phi_d \leq \phi \leq \phi_i$, where ϕ_i is another threshold, the sending rate is kept constant; when $\phi > \phi_i$ the sending rate in additively increased. The parameters used for the additively increase or multiplicatively decrease the sending rate are tuned as a function of the connection RTT in order to counteract the effect of large RTTs (Akyildiz *et al.*, 2004). This paper proposes an innovative rate-based congestion control we will refer to as Adaptive Rate Control (ARC), which has been designed as the rate-based version of classic sliding-window TCP congestion control (Jacobson, 1988; Mascolo, 1999). Finally, ARC has been implemented in the *ns*-2 simulator (Ns-2, 2002) and its performances have been compared with respect to those of New Reno, Reno Sack, Reno Fack, Vegas and Westwood+ TCP, which are the only ones having an available ns-2 implementation. We consider interplanetary scenarios with RTT up to 2000s and packet drop rate up to p=0.01. Simulation results have shown that ARC provides significant goodput improvements with respect to the other TCP flavors in the presence of RTTs larger than 1s and smaller than 2000s and packet loss probability ranging from 0.0001 to 0.01. The paper is organized as follows: Section 2 sketches the control theoretic basis of ARC; Sec. 3 describes the proposed algorithm; Section 4 shows simulation results and, finally, the last section draws the conclusions.

2. A CONTROL LAW BASED ON THE SMITH PREDICTOR: BACKGROUND RESULTS

This section summarizes some control theoretical results derived in (Mascolo, 1999) that will be used as starting points to design the ARC algorithm proposed in this paper. In (Mascolo, 1999) it has been shown that a data connection can be modelled as a time delay system that can be efficiently controlled by following the Smith principle (Astrom. and Wittenmark, 1995). In particular, to provide bottleneck queue stability and high utilization of the bottleneck link depicted in Fig. 2, the following rate-based control equation has been proposed:

$$r(t) = k[w(t) - q(t - T_{fb}) - \int_{t-RTT_{min}}^{t} r(\tau)d\tau]^{+} \quad (1)$$

where:

- $[x]^{+} = max\{0, x\}$;
- $r(t)$ is the transmission rate;
- $w(t)$ represents a threshold for the queue length $q(t)$;
- $q(t)$ is the bottleneck queue backlog;
- $RTT_{min} = T_{fw} + T_{fb}$ is the minimum round trip time, where T_{fw} is the forward delay that models the propagation time from the sender to the the bottleneck and T_{fb} is the backward delay that models the propagation time from the bottleneck to the destination and then back to the sender (see Fig. 2);
- $\int_{t-RTT_{min}}^{t} r(\tau)d\tau + q(t - T_{fb})$ represents the in pipe packets plus the queued packets, that is, they are the *outstanding packets*;
- $b(t)$ is the bandwidth used by the flow;
- k is the proportional gain that relates the transmission rate $r(t)$ to the quantity $[w(t) - q(t - T_{fb}) - \int_{t-RTT_{min}}^{t} r(\tau)d\tau]$

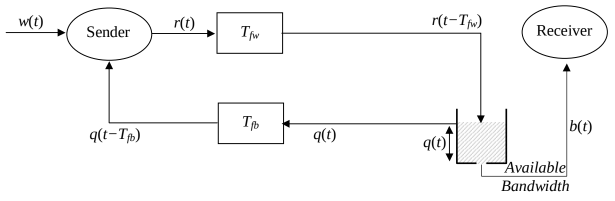

Fig. 2. Schematic of a connection.

It is easy to give an intuitive interpretation of the Eq. (1): the transmission rate $r(t)$ is proportional, via the constant k, to the difference between the threshold $w(t)$ and the sum of the backlog $q(t - T_{fb})$ with the number of in pipe packets $\int_{t-RTT_{min}}^{t} r(\tau)d\tau$ (Mascolo, 1999). From Equation (1) it turns out that when the number of outstanding packets $q(t-T_{fb}) + \int_{t-RTT_{min}}^{t} r(\tau)d\tau$ is greater or equal to w, then the computed transmission rate is zero. This implies that the number of outstanding packets can never exceed w. It is also interesting to observe that the Equation (1) can be viewed as the rate based version of the classic sliding window control. In fact, dividing both sides of Eq. (1) by k, the sliding window control equation

$$\Delta W = r/k = (w(t) - q(t - T_{fb}) - \int_{t-RTT_{min}}^{t} r(\tau)d\tau)$$

is easily obtained. Notice that if $q(t)$ is the receiver buffer queue length, then $w(t) - q(t - T_{fb})$ is the TCP advertised window and the Eq. (1) reduces to the standard TCP flow control (Mascolo, 1999;

Mascolo, 2001). If $q(t - T_{fb}) + \int_{t-RTT_{min}}^{t} r(\tau)d\tau$ represents the outstanding packets and $w(t)$ the congestion window *cwnd*, then the Eq. (1) represents the TCP congestion control.

In (Mascolo, 1999), it has been shown that Equation (1) ensures both network stability and bounded queue lengths at the routers. Moreover, by considering the control equation (1) in steady state condition, i.e. when the sending rate $r(t)$ is constant and matches the available bandwidth $b(t) = B$, and by assuming that the backlog queue length $q(t - T_{fb})$ is zero, so that the round trip time RTT reduces to the minimum round trip delay RTT_{min}, one has that $r = k \cdot (w - B \cdot RTT_{min}) = B$, from which the following relation turns out

$$w = B \cdot (RTT_{min} + \frac{1}{k}) \tag{2}$$

The Eq. (2) is important since it provides the way to set the window $w(t)$ that ensures no queue backlog when in the presence of the available bandwidth B. It should be noted that, since Eq. (2) clears out all buffers along the connection path, it improves statistical multiplexing of flows going through FIFO buffers and increases fairness in bandwidth allocation.

The constant gain k in Eq. (1) affects the time constant of the system dynamics. In fact, in (Mascolo, 1999) it has been shown that the transfer function from the threshold $w(t)$ to the queue backlog $q(t)$ is:

$$\frac{Q(s)}{W(s)} = \frac{k}{s+k} e^{-s \cdot T_{fw}} \tag{3}$$

This means that the closed-loop dynamics is first order system, with time constant $\tau = 1/k$, delayed by T_{fw}. To get a further insight into the dynamic behavior of the transfer function (3) it is worth reporting the response to the step function $w^0 \cdot 1(t)$, [1] which is

$$q(t) = w^0(1 - e^{-k(t-T_{fw})})1(t - T_{fw}) \tag{4}$$

In principle the transient mode $e^{-k(t-T_{fw})}$ in Eq. (4) can be made faster and faster by choosing a larger and larger gain k. However, an upper bound must be considered when choosing k. In fact, in packet networks the feedback information is delivered through packets, which implies that the controlled system is a sampled controlled system (Astrom. and Wittenmark, 1995). Let us assume that a feedback report is generated every round trip time, to be conservative we choose a constant time that is four times the sampling period, that is, we assume $\tau = 1/k = 4RTT_{min}$.

[1] The step function is defined as $1(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}$; w^0 is a real constant.

3. THE ADAPTIVE RATE CONTROL ALGORITHM

The congestion control algorithm we are proposing is designed as the rate-based version of the classic sliding window control algorithm used in the TCP protocol. This design choice has two fundamental motivations: the first one is to build upon the roots of an algorithm that has been extremely successful in the real world-wide Internet; the second is to provide a control algorithm that behaves friendly towards TCP. We exploit the results summarized in Section 2 for which the Eq. (1) is the rate-based form of the classic sliding window control employed by Reno TCP for flow and congestion control (Allman *et al.*, 1999; Jacobson, 1988). Thus, similarly to the TCP, the Adaptive Rate Control algorithm proposed in this paper is made of 2 phases: (1) a probing phase, which aims at utilizing the network available bandwidth; (2) a shrinking phase that reduces the input rate in the presence of congestion.

3.1 The probing phase

In order to be friendly towards Reno, we design the probing phase following the results reported in Section 2 for which the Eq. (1) is the rate-based form of the TCP flow and congestion control. Therefore we propose a *quick* probing phase, which corresponds to the TCP slow-start, and a *gentle* probing phase, which corresponds to the TCP congestion avoidance phase. The *quick* probing phase is obtained by setting

$$W(t) = W(t_0) \cdot 2^{\frac{t-t_0}{\alpha}} \tag{5}$$

where t_0 is the time of the last window update and α is a multiplicative constant. The setting (5) mimics the exponential increasing of the TCP slow start. The *gentle* probing phase is obtained by linearly increasing the control window $w(t)$ as follows:

$$W(t) = W(t_0) + \frac{t - t_0}{\alpha} \tag{6}$$

where t_0 is the time of the last window update and α is a multiplicative constant. We choose $\alpha = 0.1s$: this choice increases W of 1 packet every $100ms$ during the *gentle* probing phase, and doubles W every $100ms$ during the *quick* probing phase, i.e., it behaves as TCP over a typical terrestrial connection with RTT=100 ms.

It is important to observe that the ARC linear increasing phase mimics the linear phase used by Reno during the congestion avoidance phase. On the other hand, the additive rate increase proposed in (Rejaie *et al.*, n.d.; Kim and Bharghavan, 1999; Turletti and Huitema, 1996) *is not equivalent* to the Reno linear increasing phase (see also (Bolot and Turletti, 1998; Bansal *et al.*, 2001)).

3.2 The shrinking phase

When a congestion episode happens it is necessary to trigger the shrinking phase in order to reduce the input rate. It is important to realize that end-to-end congestion control algorithms do not explicitly know the congestion status of network nodes but they must infer it using implicit notification such as timeouts or duplicated ACKs in TCP.

We consider two events as implicit indication of congestion:

(1) a packet is lost and the sequence of received packets contains a hole;

(2) the sender does not receive any report from the receiver for a long time so that a timeout expires.

ARC reacts to congestion events by setting $w(t)$ accordingly to Equation (2), which ensures that all the buffers along the path are cleared out.

To implement the Eq. (2) it is necessary to estimate the available bandwidth. For that purpose, it should be noted that the bandwidth used at the time of a congestion episode is, by definition, the end-to-end "best effort" bandwidth available at the end of the probing phase.

To estimate the bandwidth used by a connection, the receiver counts and filters the received packets by exploiting a technique similar to that proposed in (Grieco and Mascolo, 2002).

4. PERFORMANCE EVALUATION

In this Section we investigate and compare the performance of ARC with other protocols for which there exist a *ns*-2 implementation available on line (Ns-2, 2002), i.e. with New Reno, Reno Sack, Reno Fack (Mathis and Mahdavi, 1996; Fall and Floyd, 1996), Vegas, and Westwood+ (Mascolo *et al.*, 2001; Grieco and Mascolo, 2002). For that purpose, similarly to what has been done in (Akan *et al.*, 2002), a very simple planetary scenario is considered in order to isolate the effect of packet drop rate and RTT. In particular, one persistent TCP sender transmits a 1GByte file over a 1Mbps link, which is affected by uniformly distributed random losses in both directions. The bottleneck buffer capacity is set equal to 100 packets. Unless otherwise specified, TCP sinks implement the delayed ACK option, packets are 1500 Bytes long, and connections are greedy. For each considered scenario, we run 10 simulations by varying the seed of the random loss process. All the ten measured goodputs along with the average value are reported.

4.1 The impact of the RTT

Fig. 3 reports goodputs versus the connection RTT when the inter-planetary link is affected by a packet loss rate $p = 0.001$. It shows that ARC provides full link utilization when RTT is smaller than 10s, whereas the goodput of Westwood+ TCP rolls down when RTT is larger than 2s. Regarding other considered TCP flavors, goodputs roll down when RTT is larger than 0.8s. It is worth noting that ARC provides full link utilization up to RTT=10s, whereas goodputs provided by other TCPs are smaller than 10% of the link capacity for RTT=10s. Finally, for RTT=2000s, all the TCP flavors are basically unable to send data, while ARC with Adaptive Probing provides roughly a 10kbps goodput. The reason is that the probing phase of ARC does not depend on the connection RTT as in the case of the other considered TCP flavors.

4.2 The impact of the loss rate

Fig. 4 shows goodputs as a function of the packet loss rate of the planetary link, when $RTT = 600s$. In particular, it shows that the goodput improvement due to ARC ranges from one to two order of magnitude when the packet loss ranges from 0.0001 to 0.01. The reason is that upon packet losses, ARC shrinks the control window w using the adaptive setting (2), which depends on the bandwidth estimate B, thus allowing the connection not to lose ground when in the presence of losses not due to congestion.

4.3 The impact of the file size

Fig. 5 shows the goodputs of ARC vs. the file-size, in the cases of RTT=2s,20s,200s and $p = 0.001$. The first consideration that turns out by looking at Fig. 5 is that the goodput of ARC decreases when the RTT increases (see also Fig. 3); the second one is that the goodput of ARC increases with the file-size up to a maximum value, which is reached for file sizes of roughly few tens of MBytes in the case of RTT=200s. This is an important feature,since in the paper (Akyildiz *et al.*, 2004)

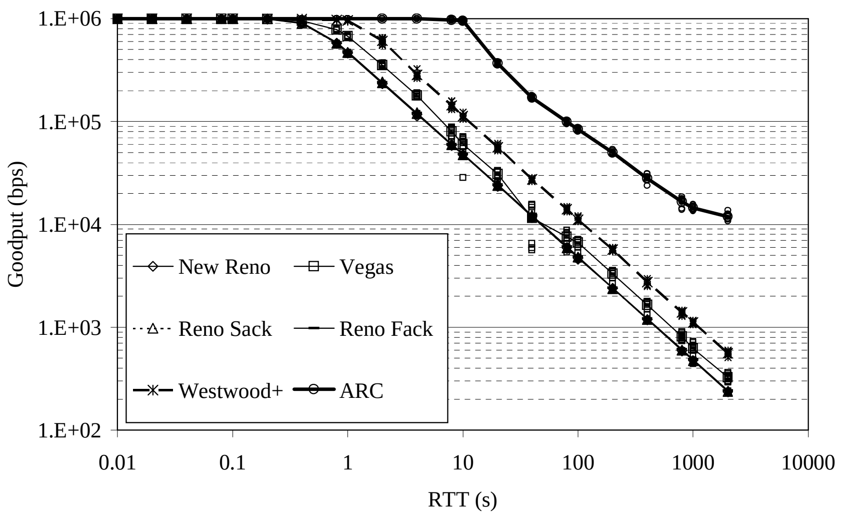

Fig. 3. Goodputs as a function of the RTT over a planetary path in the presence of a uniformly distributed loss rate p=0.001.

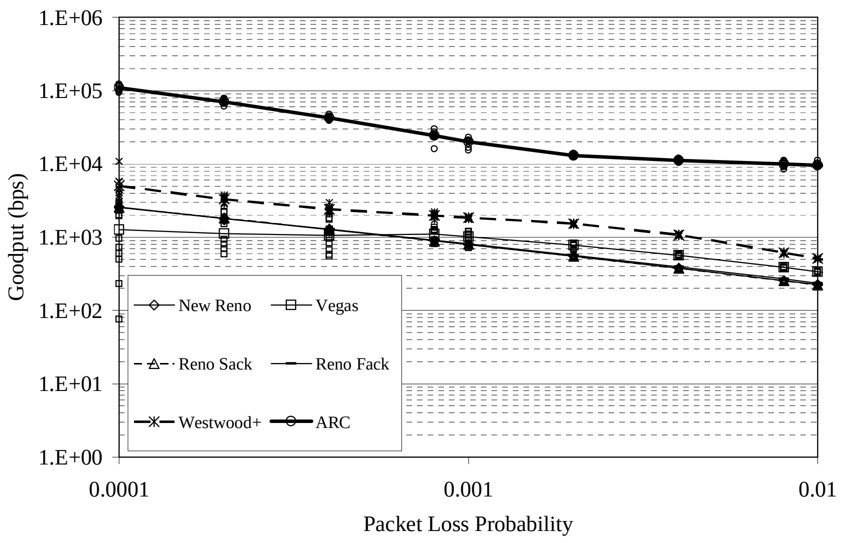

Fig. 4. Goodputs as a function of the packet loss probability over a planetary path with RTT=600s.

it has been shown that TCP-Planet reaches the maximum throughput for file-sizes larger than hundreds of Mbytes.

5. CONCLUSIONS

A rate based congestion control algorithm for planetary paths has been proposed, which has been refereed to as ARC. Using the *ns-2* simulator, it has been shown that ARC improves the goodput with respect to New Reno, Reno Sack, Reno Fack, Vegas, and Westwood+ TCP in the presence of RTTs larger than 1s and smaller than 2000s and packet loss probability ranging from 0.0001 to 0.01.

REFERENCES

Akan, O. B., J. Fang and Ian F. Akyildiz (2002). Performance of TCP Protocols in Deep Space Communication Networks. *IEEE Communications Letters.*

Akyildiz, I. F., G. Morabito and S. Palazzo (2001). TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks. *IEEE/ACM Transactions on Networking* **9**(3), 307–321.

Akyildiz, I. F., O. B. Akan and J. Fang (2004). TCP-Planet: A Reliable Transport Protocol for InterPlaNetary Internet. *Journal on selected Areas in Communications* **22**(2), 348–361.

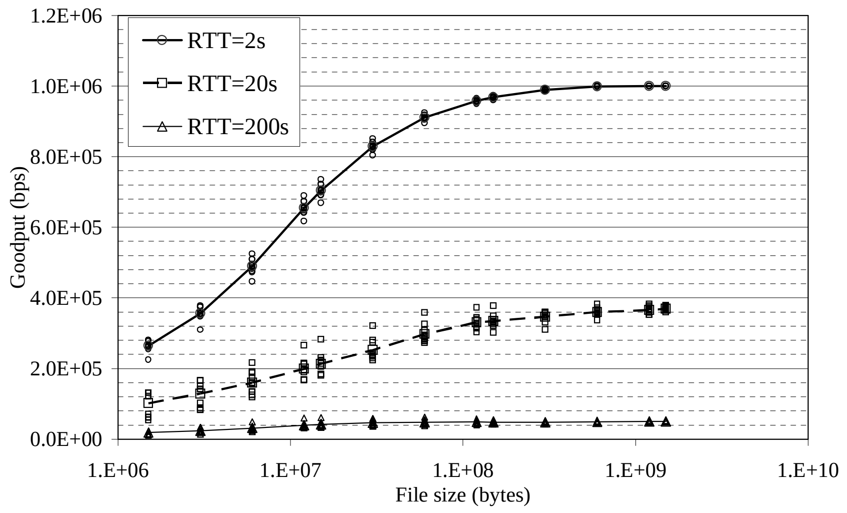

Fig. 5. Goodputs of ARC as a function of the file size in the presence of a uniformly distributed loss rate p=0.001.

Allman, M., V. Paxson and W. R. Stevens (1999). TCP congestion control. RFC 2581.

Astrom., K. J. and B. Wittenmark (1995). *Computer controlled systems: theory and design.* Prentice Hall Information and System Sciences serie. 3 ed.. Prentice Hall, Englewood Cliffs.

Bansal, D., H. Balakrishnan, S. Floyd and S. Shenker (2001). Dynamic behavior of slowly-responsive congestion control algorithms. In: *ACM SIGCOMM 2001.* UC San Diego, CA, USA. pp. 263–273.

Bolot, J. C. and T. Turletti (1998). Experience with control mechanisms for packet video in the internet. *ACM SIGCOMM Computer Communication Review* **28**, 4–15.

Fall, K. and S. Floyd (1996). Simulation-based Comparisons of Tahoe, Reno, and SACK TCP. *Computer Communication Review.*

Grieco, L. A. and S. Mascolo (2002). Westwood TCP and easy RED to improve fairness in high speed networks. In: *IFIP/IEEE Seventh International Workshop on Protocols For High-Speed Networks.* Berlin, Germany. pp. 130–146.

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

Jacobson, V. (1988). Congestion avoidance and control. In: *ACM Sigcomm '88.* Stanford, CA, USA. pp. 314–329.

Kim, T. and V. Bharghavan (1999). Improving congestion control performance through loss differentiation. In: *International Conference Computer and Communications Networks.* Boston, MA.

Mascolo, S. (1999). Congestion control in high-speed communication networks using the Smith principle. *Automatica, Special Issue on Control methods for communication networks* **35**, 1921–1935.

Mascolo, S. (2001). Insights into TCP/IP congestion control using the Smith principle. In: *European Control Conference.*

Mascolo, S., C. Casetti, M. Gerla, M. Sanadidi and R. Wang (2001). TCP westwood: End-to-end bandwidth estimation for efficient transport over wired and wireless networks. In: *ACM Mobicom 2001.* Rome, Italy.

Mathis, M. and J. Mahdavi (1996). Forward acknowledgement: Refining TCP congestion control. In: *ACM SIGCOMM '96.* Stanford CA.

Ns-2 (2002). network simulator.

Rejaie, R., M. Handley and D. Estrin (n.d.). RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the internet. In: *IEEE Infocom 1999.* New York. pp. 1337–1345.

Turletti, T. and C. Huitema (1996). Videoconferencing on the internet. *IEEE/ACM Transaction on Networking* **4**(3), 340–351.