



Open Archive Toulouse Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in: <http://oatao.univ-toulouse.fr/>
Eprints ID: 6238

To cite this document: Diana, Rémi and Lochin, Emmanuel *Analytical Model of TCP Relentless Congestion Control*. (In Press: 2012) In: 6th International Workshop on Verification and Evaluation of Computer and Communication Systems, 27-28 Aug 2012, Paris, France.

Any correspondence concerning this service should be sent to the repository administrator: staff-oatao@inp-toulouse.fr

Analytical Model of TCP Relentless Congestion Control

Rémi Diana

Université de Toulouse
F-31077 Toulouse, France
ISAE-TeSA / CNES / Thales Alenia Space
remi.diana@isae.fr

Emmanuel Lochin

CNRS ; LAAS ; 7 avenue du colonel Roche
F-31077 Toulouse, France
Université de Toulouse
UPS, INSA, INP, ISAE ; LAAS
F-31077 Toulouse, France
emmanuel.lochin@isae.fr

We introduce a model of the Relentless Congestion Control proposed by Matt Mathis. Relentless Congestion Control (RCC) is a modification of the AIMD (Additive Increase Multiplicative Decrease) congestion control which consists in decreasing the TCP congestion window by the number of lost segments instead of halving it. Despite some on-going discussions at the ICCRG IRTF-group, this congestion control has, to the best of our knowledge, never been modeled. In this paper, we provide an analytical model of this novel congestion control and propose an implementation of RCC for the commonly-used network simulator ns-2. We also improve RCC with the addition of a loss retransmission detection scheme (based on SACK+) to prevent RTO caused by a loss of a retransmission and called this new version RCC+. The proposed models describe both the original RCC algorithm and RCC+ improvement and would allow to better assess the impact of this new congestion control scheme over the network traffic.

Relentless Congestion Control, Performance Evaluation

1. INTRODUCTION

Relentless Congestion Control (RCC) is a proposal from Matt Mathis which consists in a simple modification of the AIMD (Additive Increase Multiplicative Decrease) congestion control algorithm (7). Basically, instead of halving the TCP congestion window after a loss, RCC decreases the current congestion window by the number of lost segments. This behaviour can be modeled as a strict implementation of van Jacobson's Packet Conservation Principle (this principle suggests that a new packet should not be placed into the network until an old packet leaves). Indeed, during recovery, new segments are injected into the network in exact accordance with those that have been delivered to the receiver (8).

RCC is not an AIMD-friendly protocol and as a result, requires that the network allocates capacity through Fair Queuing or Fair Dropping queue management (2). RCC could perform efficiently over network architectures that enable Quality of Service (QoS) guarantees such as (3). Indeed, a decade of research in QoS has shown that the standard TCP reaction to congestion events (which is to halve its congestion window) can be counterproductive over these QoS networks (4),

(6). The principle to decrease the current congestion window by the number of lost segments can also be implemented inside current congestion controls such as CUBIC, Newreno or Compound. Intuitively, RCC might enhance the performance of standard congestion controls when losses are not due to congestion (over very noisy wireless links) or large bandwidth-delay product networks (LBDP). In the first case, RCC would prevent large congestion window decrease due to error link losses while in the context of LBDP networks and long delay links, RCC might achieve a higher throughput. However, whatever the context of use, there is no existing analytical model of RCC allowing to estimate the expected rate that would achieve this protocol over a given network. In order to clearly assess the impact of a deployment of such TCP modification, it is obviously essential to model this TCP variant.

In this paper we introduce two models. The first one presented in Section 2 does not take RTO into account. Our second model described in section 3 is an extension that integrates the RTO effects on congestion window evolution. To prevent RTO triggering, we combine RCC algorithm with a lost retransmission detection scheme based on SACK+ (5). Thus, we implemented in ns-2 an improvement of RCC algorithm that we called RCC+

corresponding to our first model. Our second model corresponds to the original RCC algorithm that do not avoid RTO (also implemented inside ns-2). We present RCC+ in Section 4 and evaluate the accuracy of both models in Section 5 with ns-2 simulations. Finally, we conclude this work in Section 7.

2. ANALYTICAL MODEL

In this section, we develop a stochastic model of TCP Relentless Congestion Control algorithm coupled with the algorithm of selective acknowledgement (SACK). This leads to a simple analytic expression for the throughput of a TCP Relentless sender as a function of loss rate p and the average round trip time (RTT). Our RCC model is built on the well-known Padhye *et al.* TCP model (9) from which we borrow the notations and the scheme given in Fig. 1 to ease the understanding.

The notations are presented in Fig. 1. The period denoted TD defines an elementary cycle (corresponding to TD_P in (9)), delimited by two consecutive decreases of the window W_i where i refers to the i^{th} TD .

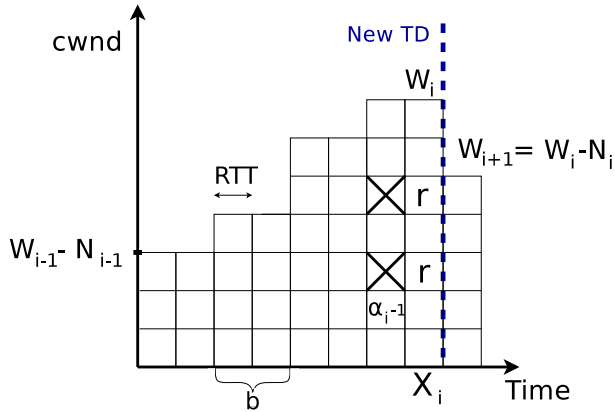


Figure 1: Evolution of the congestion window size over time.

In the model, we adopt the following rule for all random variables: V_i corresponds to the value of the random variable V in TD_i and its expected value computed on all TD_i is noted $E[V]$.

Let Y_i be the number of packets sent in TD_i ; α_i the index of the first lost packet; β_i the amount of packets sent after α_i to complete the round (corresponding to an RTT) and N_i the number of retransmissions done in TD_i . We also define X_i the number of rounds in TD_i and $\frac{1}{b}$ the acknowledgement generating frequency. Thus:

$$Y_i = \alpha_i + \beta_i + W_i \quad (1)$$

and

$$E[Y] = E[\alpha] + E[\beta] + E[W] \quad (2)$$

We now have to derive $E[\alpha]$ and $E[\beta]$. If we consider uniform and independent losses, $\alpha_i = k$ means that the first $k - 1$ packets are successfully sent and the k^{th} packet is lost. The packet loss probability is denoted p . We can compute $E[\alpha]$ as follows:

$$P(\alpha = k) = (1 - p)^{k-1}p \longrightarrow E[\alpha] = \sum_{k=1}^{\infty} (1 - p)^{k-1}p \cdot k$$

$$E[\alpha] = \frac{1}{p} \quad (3)$$

β_i evolves from 1, if the losses occur at the end of the window, to $W_i - \frac{1}{b}$, if losses occur at the beginning of the window. The uniform aspect of losses implies the uniform distribution of β_i in $[1, W_i - \frac{1}{b}]$. It follows that:

$$E[\beta] = \frac{W_i + 1 - \frac{1}{b}}{2} \simeq \frac{E[W] + 1 - \frac{1}{b}}{2} \quad (4)$$

The relation between window size in TD_{i-1} and TD_i can be written as follows:

$$W_i = W_{i-1} - N_{i-1} + \frac{X_i - 1}{b} \quad (5)$$

as a consequence:

$$E[N] = \frac{E[X] - 1}{b} \quad (6)$$

The expected value of N can be evaluated with the same assumptions. Basically, if we consider uniform and independent losses with the elementary probability of p , then N follows a binomial law of parameters p and $E[\beta]$ (the mean amount of packets sent after the first loss occurs). Thus, $E[N] = 1 + p(E[\beta])$ and (6) leads to:

$$E[X] = 1 + b \cdot E[N] = 1 + b \left(1 + p \frac{E[W] + 1 - \frac{1}{b}}{2} \right) \quad (7)$$

The evolution of the window size can also be written using slope $\frac{1}{b}$, which corresponds to the evolution pace of W .

$$Y_i = \sum_{k=0}^{X_i-1} \left(W_{i-1} - N_{i-1} + \frac{k}{b} \right)$$

$$= X_i \left(W_{i-1} - N_{i-1} - \frac{1}{2b} \right) + \frac{X_i^2}{2b} \quad (8)$$

If we now take the mathematical expectation of (8) assuming for a first approximation that X_i and W_i

are mutually independent sequence of i.i.d. random variables and with $V[X]$ the variance of X we have:

$$E[Y] = E[X] \left(E[W] - E[N] - \frac{1}{2b} \right) + \frac{E[X]^2}{2b} + \frac{V[X]}{2b} \quad (9)$$

If we combine (9), (2), (3), (4) and (7) we obtain:

$$\begin{aligned} & \frac{bp(4-p)}{8} E[W]^2 + \left(b - \frac{1}{2} + \frac{p^2(1-b)-3p}{4} \right) E[W] - \frac{p+1}{p} \\ & + \frac{p(-2b^2+b+1)}{4b} - \frac{b^2+1}{2b} + \frac{V[X]}{2b} + \frac{p^2(b^2-1)}{8b} = 0 \quad (10) \end{aligned}$$

By solving (10) and keeping only the positive root, we obtain a literal expression of $E[W]$ (we do not provide this long expression which is out of interest here). Moreover, the duration of TD_i , A_i , can be expressed as the number of rounds in TD_i , X_i , multiplied by the average duration of a round, RTT : $A_i = X_i \cdot RTT$. Thus, $E[A] = E[X] \cdot RTT$. Now, if we consider p near zero and combine the expressions of $E[W]$, $E[Y]$ and $E[A]$, we obtain the average throughput of a Relentless flow T_p :

$$T_p = \frac{E[Y]}{E[A]} = \frac{MSS}{b \cdot RTT \cdot p} + o\left(\frac{1}{p}\right) \quad (11)$$

where MSS is the maximum segment size. Let C , be the constant term in (11). We have a general expression for the throughput of a Relentless flow which is:

$$T_p = \frac{C \cdot MSS}{RTT \cdot p} + o\left(\frac{1}{p}\right), \quad C = \frac{1}{b} \quad (12)$$

TCP retransmission time out (RTO) is not considered in this model. Nevertheless, by combining RCC and SACK retransmission scheme, we are mainly sensitive to one type of RTO which is the loss of a retransmitted packet. If the loss rate is low, this event can occur but can be considered as rare. However, we present in the next section an improvement of RCC allowing to prevent RTO due to the loss of a retransmitted packet. As a result, the model developed here allows to correctly fit our current implementation (presented in the following Section 5) and motivates why we do not need to take into consideration the RTO.

3. ANALYTICAL MODEL WITH RTO

In this section, we further develop our previous model to handle the case of RTO. We adopt a different approach following the congestion window evolution which behaves as shown in Fig. 2. One of the main

challenge of this model is that compared to the standard halving congestion window scheme (9), we must take into account the number of lost packets during a round. This number, obtained following a probability model, greatly complexifies the model.

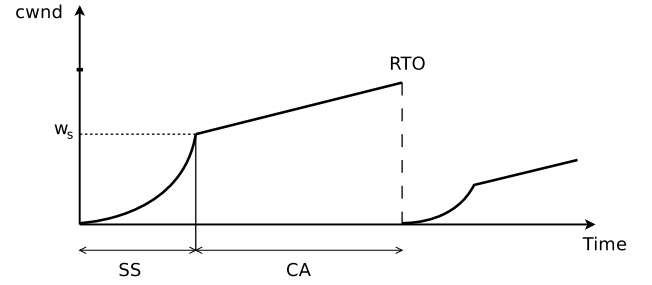


Figure 2: Evolution of the congestion window size over time in case of RTO.

The evolution of the congestion window can be described as a repetition of slow start (SS) and congestion avoidance (CA) phases. The SS phase is the same as the one present in TCP Reno or Newreno. This phase is left once the congestion window reaches a threshold or when a loss is detected. In Figure 2, w_s represents the value of the congestion window at the end of this phase. The second phase of the cycle corresponds to the phase previously modelled (Section 2) in which the window is increased by one minus the number of losses after each RTT.

Let X_{SS} , X_{CA} be respectively the number of packets sent during the SS and CA phases and D_{SS} , D_{CA} the respective duration of SS and CA phases. Throughout in case of RTO, T_{RTO} , can still be expressed as:

$$T_{RTO} = \frac{E[X_{SS}] + E[X_{CA}]}{E[D_{SS}] + E[D_{CA}]} \quad (13)$$

In the following, we develop each of the terms of (13):

$$E[X_{SS}] = \sum_{k=1}^{lim} k \cdot P(X_{SS} = k) \quad (14)$$

In (14), lim represents that X_{SS} increases until the reach of a threshold or when a loss occurs. As losses are considered to be uniformly distributed, we consider in a first approximation, lim to be in average equals to $\frac{1}{p}$.

To express the amount of sent packets, we consider the congestion window in the CA phase as an arithmetico-geometric sequence denoted $(W_{CA}^n)_{n \in \mathbb{N}}$. After an RTT, the congestion window is increased by one and decreased by the number of losses which depends on the size of the previous congestion window value. Indeed, as the number of losses follows a binomial law of parameters p and W_{CA}^n , the average number of losses is pW_{CA}^n . Thus, the first term of this sequence is w_s and the evolution of W_{CA} is given by $W_{CA}^{n+1} = W_{CA}^n + 1 - pW_{CA}^n$. The

general term of this sequence can be written as follows:

$$W_{CA}^n(w_s) = (1-p)^n(w_s - \frac{1}{p}) + \frac{1}{p} \quad (15)$$

Let I_{RTO} be the index of the RTO event in the CA phase. I_{RTO} evolves from 1 to $+\infty$. $E[X_{CA}]$ depends on w_s and is given by:

$$E[X_{CA}](w_s) = \sum_{k=1}^{\infty} [P(I_{RTO} = k) \cdot \left(\sum_{j=1}^k W_{CA}^j(w_s) - \frac{W_{CA}^k(w_s)}{2} \right)] \quad (16)$$

In (16), the term $-\frac{W_{CA}^k(w_s)}{2}$ represents the fact that when an RTO occurs, the end of the window is lost. In average, we can consider that the RTO occurs in the middle of the window leading to the loss of the second half of the window. To trigger an RTO, a packet and its retransmission have to be lost. The probability of this event is p^2 . RTO can occur for any packet of the window. As a consequence:

$$P(I_{RTO} = k) = p^2 W_{CA}^{k-1}(w_s) \quad (17)$$

Moreover, as the value of the congestion window is equal to the amount of packets sent during SS phase, w_s can evolve from 1 to lim . Thus we have:

$$P(w_s = k) = (1-p)^{k-1}p$$

and (16) leads to:

$$E[X_{CA}] = \sum_{i=1}^{lim} \left[p(1-p)^{i-1} \sum_{k=1}^{\infty} \left(p^2 W_{CA}^{k-1}(i) \left(\sum_{j=1}^k W_{CA}^j(i) - \frac{W_{CA}^k(i)}{2} \right) \right) \right] \quad (18)$$

We now focus on the expression of phases duration. During the SS phase, the amount of sent packets increases as of power of 2. As a consequence, $E[D_{SS}]$ is given by:

$$\begin{aligned} E[D_{SS}] &= \sum_{k=1}^{lim} \log_2(k) P(X_{SS} = k) \\ &= \sum_{k=1}^{lim} \log_2(k) (1-p)^{k-1}p \end{aligned} \quad (19)$$

and

$$\begin{aligned} E[D_{CA}](w_s) &= \sum_{i=1}^{\infty} i P(I_{RTO} = i) \\ &= \sum_{i=1}^{\infty} i \cdot p^2 W_{CA}^{i-1}(w_s) \end{aligned} \quad (20)$$

As a consequence the global average of D_{CA} is:

$$\begin{aligned} E[D_{CA}] &= \sum_{j=1}^{lim} P(w_s = j) E[D_{CA}](w_s) \\ &= \sum_{j=1}^{lim} p(1-p)^{j-1} \left(\sum_{i=1}^{\infty} i \cdot p^2 W_{CA}^{i-1}(j) \right) \end{aligned} \quad (21)$$

Finally, combining (13), (14), (17), (18), (19), (21) we obtain:

$$T_{RTO} = \frac{\Theta}{\sum_{j=1}^{lim} \left((p(1-p)^{j-1}) (\log_2 j + \sum_{i=1}^{\infty} i \cdot p^2 W_{CA}^{i-1}(j)) \right)} \quad (22)$$

with:

$$\Theta = \sum_{i=1}^{lim} \left[p(1-p)^{i-1} \left(i + \sum_{k=1}^{\infty} \left(p^2 W_{CA}^{k-1}(i) \left(\left(\sum_{j=1}^k W_{CA}^j(i) - \frac{W_{CA}^k(i)}{2} \right) \right) \right) \right) \right]$$

Numerically, we observe that $E[X_{CA}](w_s)$ does not depend on w_s leading to the following approximation for T_{RTO} :

$$T_{RTO} \simeq \frac{C_{RTO} MSS}{RTT \cdot p} + o\left(\frac{1}{p}\right) \quad \text{with } C_{RTO} = 0.49$$

We have verified that the last approximation is closed to T_{RTO} and thus considers in the following this latest and simplified expression.

4. RCC+ IN A NUTSHELL

As basic RCC algorithm with SACK mechanism is not RTO resistant, we propose to implement RCC with SACK+ (5) in ns-2. SACK+ is a SACK extension allowing to prevent RTO due to the loss of retransmitted packets.

We have implemented an improvement of RCC in order to prevent RTO caused by a loss of a retransmission. Of course, if a packet is lost each time it is retransmitted RTO cannot be avoided. For new connection requests, the retransmission timer is initialized to 3 seconds (1). By default, a segment with the SYN flag set, is resent no more than three times. In other words, it must be received before four RTTs elapse, which is approximately the value of the retransmission timer. The configuration of this timer is also possible to enable more retransmission tries.

In our implementation, we choose a discrete resolution of the lost retransmission problem as in (5). However,

we could choose a continuous solution and set a timer to each retransmitted packet. We could set this timer to $k * RTT$, with RTT the current estimation of RTT and $k > 1$ to prevent spurious retransmissions.

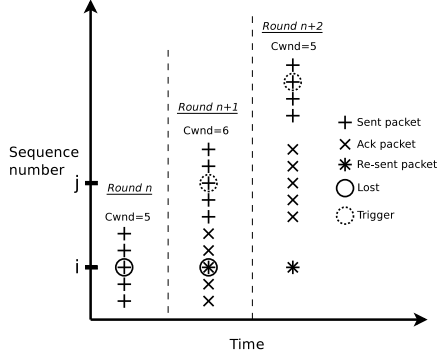


Figure 3: $RCC+$ behaviour

In our case, for each retransmitted packet, we fix a trigger. This trigger corresponds to the acknowledgement of the following *regular* packet which is sent after the retransmission. By *regular*, we refer to packets that do not correspond to a retransmission. Actually, if the regular packet sent after the retransmission is acknowledged before the retransmitted one, we can suppose that the retransmitted packet is lost. As shown in Fig. 3, when packet i is retransmitted in round $(n + 1)$, the following regular packet sent which is packet j , is used as a trigger for an eventual new retransmission. Indeed, if packet j is acknowledged and not packet i , packet i is considered as lost and re-emitted. In this example, as the first retransmission of packet i is lost, packet j is acknowledged and not packet i . This leads to a second retransmission of packet i in round $(n + 2)$ and the initialisation of a new trigger. Anyway, all these solutions are only different by their implementation. In a general manner, the result remains the same: RTO due to loss of retransmission are avoided.

5. IMPLEMENTATION AND EVALUATION OF RCC AND $RCC+$

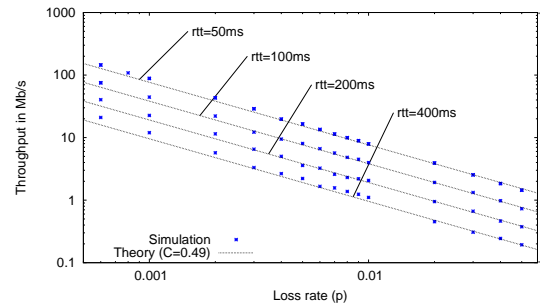
Our first model without RTO consideration corresponds to $RCC+$ implementation as detailed in Section 4. The complete model presented in Section 3 corresponds to the basic implementation of RCC algorithm. In this part we first want to demonstrate the accuracy of our models and underline the interest brought by $RCC+$.

Our models give simple approximations of the throughput of an RCC flow as a function of the RTT and the loss rate p .

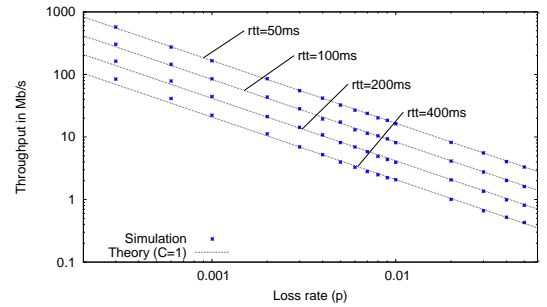
The results obtained with the ns-2 implementation of RCC are presented in Fig. 4(a) and those obtained with

the ns-2 implementation of $RCC+$ are presented in Fig. 4(b)

In Fig. 4, each point represents a simulation result with the corresponding parameters. The duration of the simulations is long enough to reach the steady state and the computation of the throughput is done when this steady state is reached. To verify the parameters of the model, simulations are done with a loss rate ranging from 0.03% to 5%. To estimate the maximum throughput of the RCC flow, we ensure that we are not limited by the link capacity between the source and the destination. Thus, we set this capacity to $100Gb/s$ which is much more than the maximum achievable theoretical throughput for the chosen parameters.



(a) RCC algorithm



(b) $RCC+$ algorithm

Figure 4: Comparison of theoretical and simulated throughput of a Relentless flow

Fig. 4 shows that the model fits the simulation with $C = 1$ which is the theoretical value of C . Note that we have verified inside the simulation traces that there was no retransmission timeout and confirmed that $RCC+$ correctly prevents RTO due to loss of retransmissions.

6. CONSTANT VALUE AND LOSS DISTRIBUTION

In Section 5 and in our models, we have considered uniform losses pattern. We now propose to investigate the value of the constant of the models in the case of bursty loss channel. We drive a set of experiments to evaluate C and C_{RTO} with an average burst size

(denoted B) ranging from 2 to 4 following a Gilbert-Elliott channel. An important point is that RCC flow throughput still evolves in $\frac{1}{p}$ even with bursty losses. Table 1 gives the results obtained.

Table 1: *Impact of loss model on C*

Loss Model		RCC+
Uniform		$C = 1.0$
Gilbert-Elliott	$B = 2$	$C = 0.90$
	$B = 3$	$C = 0.90$
	$B = 4$	$C = 0.90$

As RCC+ prevents RTO triggering, the constant remains stable making the model robust (at least up to $B = 4$). However, the slight decrease of the constant value (from 1.0 to 0.9) might be explained by the increase of the average number of losses at the end of the congestion window. Indeed, if a loss occurs at the end of the congestion window, its detection is possible only two rounds after. As the pace of the sent packets is driven by the pace of the received acknowledgements, this implies that the number of packets sent in the round following this loss is lower than the window size value. More generally, we can state that, if there are n losses at the end of the congestion window, the number of packets sent in the next TD is reduced by n .

7. CONCLUSION AND FUTURE WORK

We have proposed a model of the Relentless Congestion Control algorithm and an ns-2 implementation (available for download at <http://personnel.isae.fr/remi-diana>) based on SACK+. We confirm that RCC evolves in $\frac{1}{p}$ and our performance evaluation shows the need to enable a fair-queuing algorithm to prevent unfairness between other TCP variants.

As a next step, we propose to further assess the benefit of using RCC as a potential solution for long delay link and satellite communications. We also expect to use this model in a larger performance evaluation study which aims at evaluating RCC+ with various TCP variants.

8. ACKNOWLEDGEMENTS

This study has been supported by funding from CNES and Thales Alenia Space.

9. REFERENCES

- [1]R. Braden. Requirements for Internet Hosts - Communication Layers. RFC 1122 (Standard), 1989.
- [2]A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. *SIGCOMM Comput. Commun. Rev.*, 1989.

- [3]F Garcia, C Chassot, A Lozes, M Diaz, P Anelli, and E Lochin. Conception, implementation and evaluation of a qos - based architecture for an ip environment supporting differentiated services, 2001.

- [4]G. Jourjon, E. Lochin, and P. Sénac. Design, implementation and evaluation of a qos-aware transport protocol. *Comput. Commun.*, 2008.

- [5]B. Kim, D. Kim, and J. Lee. Lost retransmission detection for TCP SACK. *IEEE Communications Letters*, 2004.

- [6]E. Lochin and P. Anelli. Tcp throughput guarantee in the diffserv assured forwarding service: what about the results? *Annals of Telecommunications*, 2009.

- [7]M. Mathis. Relentless congestion control. In *PFLDNet*, 2009.

- [8]M. Mathis. Relentless congestion control. Internet Draft draft-mathis-iccr-relentless-tcp-00.txt, Internet Engineering Task Force, 2009. (Work in progress).

- [9]J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. *SIGCOMM Comput. Commun. Rev.*, 1998.