ANALYSIS OF TCP ALGORITHMS IN THE RELIABLE IEEE 802.11b LINK

Lukas Pavilanskas
Department Of Telecommunications
Vilnius Gediminas Technical University
Naugarduko 41, Vilnius, LT-03227, Lithuania
E-mail: lukas.pavilanskas@el.vtu.lt

KEYWORDS
Modeling, TCP, IEEE 802.11, Reliable

ABSTRACT

IEEE 802.11b technology networks have become increasingly common and an increasing number of devices are communicating with each other over lossy links. Unfortunately, TCP performs poorly over lossy links as it is unable to differentiate the loss due to packet corruption from that due to congestion.

This paper presents systematic analysis and modeling results of TCP Reno, Vegas and Veno algorithms in the IEEE 802.11b networks link. In this work the following questions are analyzed: principles of operations, and the throughput dependence on random loss and long links delays, that emerge as using from MAC layer reliable protocols.

The aim of the presented modeling is to sift features of TCP protocol algorithms and find its dependence on throughput in the IEEE 802.11 networks.

INTRODUCTION

The major problem in designing IEEE 802.11a/b/c networks is random packet losses (Lakshman and Madhow 1997) that emerge because of low signal-noise ratio. The packet losses mainly influence the throughput of the information transmitted by TCP protocol (Kumar 1998).

TCP „de-facto“ is a reliable connection-oriented protocol which implements the flow control of heterogeneous networks by means of a sliding window and „slow start“ algorithms (RFC 793) (RFC 2581) (RFC 2582). These flow control algorithms are analyzed in great detail; moreover, the models of these algorithms operating in the lossy wireless links are composed and analyzed (Jacobson and Karels 1988) (Balakrishnan et al. 1997) (Brakmo et al. 1994) (Jeonghoon et al. 1999) (Floyd and Jacobson 1993).

The most often used models of TCP algorithms i.e. OldTahoe, Tahoe, Reno, and NewReno were analyzed by A. Kumar. The author (Kumar 1998) has presented the throughput function dependence upon the probability of random losses, when in the wireless networks the non-correlative random losses operate. The results allow making a conclusion that if the probability of random losses increases the TCP session capacity reduces. But this inference cannot be applied to TCP Vegas because, if compared with Reno algorithm, TCP Vegas can raise its throughput to approximately 70% at random losses (Jeonghoon et al. 1999). Although it is possible if no other algorithms only TCP Vegas is used on the same network (Hengartner et al. 2000). TCP Veno does not have this drawback. When used on the wireless network with random packet loss rate of 1 %, this algorithm can increase throughput to 80 % (Fu and Liew 2003).

In order to prevent the random pocket losses on the wireless networks the MAC layer protocols can be used (Karl et al. 2004). The main characteristic of these protocols is that the packet is repeatedly being sent till the ACK packet of MAC layer is received. If the time needed to transmit packet takes to long, TCP algorithms detect packet loss after the timeout (Jacobson and Karels 1988). This situation is analyzed in the satellite networks links. In these links the stable delay can be up to 300 ms (Peng et al. 2001) (Partridge and Shepard 1997) (Ghani and Dixit 1999). Whereas, in the IEEE 802.11 networks when reliable protocols are used the time-span of delay is longer and random.

This paper deals with the systematic analysis and modeling results of TCP Reno, Vegas and Veno algorithms in the IEEE 802.11b networks link. The dependence of TCP session throughput on random packet losses and lengthy delays emerging as link layer reliable protocols are used, is analyzed. The modeling results demonstrate that if random packet losses are very frequent the TCP Veno algorithm should be used, and if random delay is lengthy the TCP Reno should be used.

The aim of the modeling is to analyze the features of TCP algorithms and fond out their effect on throughput in the IEEE 802.11 networks.

TCP RENO ALGORITHM

Internet is a heterogeneous system, where the packets are lost because of the different (throughput, asymmetric capacity, and etc.) characteristics of the links (Fu and Liew 2003). To avoid such losses the TCP protocol which make reliable connection with flow control is used. There are many flow control algorithms of TCP, but the TCP Reno is the mostly used (Kumar 1998).

Reno can detect congested packet losses (Jacobson and Karels 1988); however packet losses in the wireless networks are random and result by low signal-noise ratio. Therefore, the throughput decrease during the transmission carried out by the TCP protocol (Lakshman and Madhow 1997) (Fu and Liew 2003) (Balakrishnan et al. 1997).

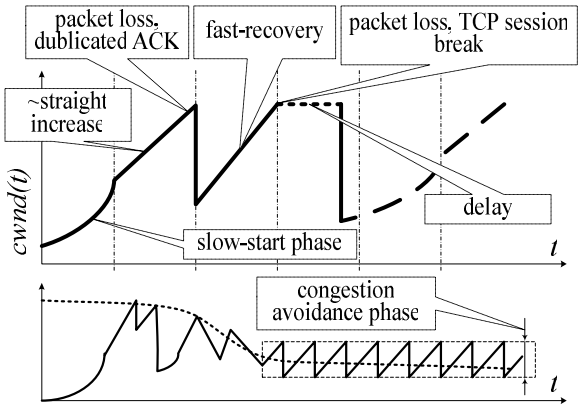
The TCP Reno algorithm realizes the flow control through the congestion window, “slow start”, and congestion control mechanisms (RFC 2581) (RFC

Proceedings 12th International Conference ASMTA 2005
Khalid Al-Begain, Gunter Bolch, Miklos Telek © ECMS, 2005
ISBN 1-84233-112-4 (Set) / ISBN 1-84233-113-2 (CD)

2582). Normally, the size of the congestion window is being periodically increased till it reaches its maximal window size ($cwnd_{\max}$). Reno uses two dynamic modes of the window: slow-start phase and congestion avoidance phase (Fig. 1.). At the moment of $t + t_A, [s]$ congestion window $cwnd(t)$ is changing like this:

$$cwnd(t + t_A) = \begin{cases} \text{"slow start" phase:} \\ \text{if } cwnd(t) < ssth(t), \text{ then} \\ cwnd(t) + 1, \\ \text{congestion avoidance phase:} \\ \text{if } cwnd(t) \geq ssth(t), \text{ then} \\ cwnd(t) + \dfrac{1}{cwnd(t)}. \end{cases} \quad (1)$$

There $ssth(t), [\text{in packets}]$ – "slow start" threshold, which indicates the TCP shift from the "slow start" to the congestion avoidance.



Figures 1: Operating Scheme of the TCP Reno Algorithm

Only after the confirming ACK packet is received, TCP detects that the packet is successfully transmitted. The threshold time during which the confirming ACK packet is received is called *RTO*:

$$RTO = \overline{RTT} + 4\sigma_{\overline{RTT}}. \quad (2)$$

There $\overline{RTT}$ – the average of the time needed to successfully transmit the packet; $\sigma_{\overline{RTT}}$ – the root-mean-square of deviation of $\overline{RTT}$ duration.

If *RTO* is too long the detection of congested packet loss will be delayed, therefore the session capacity will be reduced (RFC 2581). If during this period ACK is not received, congested packet loss is detected by TCP Reno (Fig. 1.). The congestion window becomes minimal and $ssth(t)$ decrease:

$$cwnd(t) = 1; \quad (3)$$

$$ssth(t) = \frac{cwnd(t)}{2}. \quad (4)$$

However, if the duplicated ACK of the next packet is received, the fast-recovery mechanism goes off (RFC 2581). After that the TCP shifts into the congestion avoidance phase:

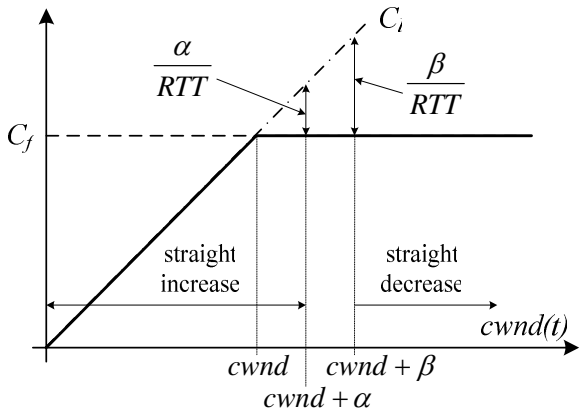
$$cwnd(t) = ssth(t); \quad (5)$$

$$ssth(t) = \frac{cwnd(t)}{2}. \quad (6)$$

If information is transmitted via the wireless link, where the packets are lost at random, the minimal congestion window size will be periodically adjusted (3). Therefore, the capacity of TCP session will decrease.

TCP VEGAS AND VENO ALGORITHMS

In the TCP Vegas algorithm the more effective congestion avoidance mechanism is realized, because $cwnd(t)$ depends on *RTT* (Hengartner et al. 2000).



Figures 2: The Congestion Control Operating Scheme of the TCP Vegas Algorithm

In Vegas algorithm (Brakmo et al. 1994), the sender measures the so-called expected and actual session capacities (Fig. 2.):

$$C_l = cwnd(t) / RTT_{\min}; \quad (7)$$

$$C_f = cwnd(t) / RTT. \quad (8)$$

There C_l – expected session capacity; C_f – actual session capacity; $RTT_{\min}$ – the minimum of measured round-trip times. Hence the difference of the rate is:

$$\Delta = C_l - C_f. \quad (9)$$

When $RTT > RTT_{\min}$, there is a bottleneck in the link where the packets of the connection accumulate (Fig. 2.) (Jeonghoon et al. 1999). Let the backlog at the queue to be denoted by *N*. Therefore, $cwnd(t)$ will decrease:

$$RTT = RTT_{\min} + \frac{N}{C_f}; \quad (10)$$

$$N = C_l (RTT - RTT_{\min}) = \Delta RTT_{\min}. \qquad (11)$$

Thus, Vegas attempts to keep N as small as it is possible by adjusting the TCP window size, and consequently avoiding packet loss due to buffer overflow (Brakmo et al. 1994). Furthermore, in contrast to TCP Reno, it will not detect $cwnd(t) = 1$ immediately irrespective to the errors that challenged the loss of the packet (random or congestion). Thus the throughput of algorithm Vegas is better approximately by 70 % if compared to TCP Reno. In this case mathematical expression of TCP Vegas is as follows:

$$cwnd(t) = \begin{cases} cwnd(t) + 1, & \text{if} \quad \Delta < \dfrac{\alpha}{RTT_{\min}} \\ cwnd(t), & \dfrac{\alpha}{RTT_{\min}} \le \Delta \le \dfrac{\beta}{RTT_{\min}}. \\ cwnd(t) - 1, & \text{if} \quad \dfrac{\beta}{RTT_{\min}} < \Delta \end{cases} \qquad (12)$$

There α – $cwnd(t)$ increase coefficient: if $cwnd_{\max} > cwnd(t) + \alpha$, then $cwnd(t) + 1$; β – $cwnd(t)$ decrease coefficient: if $cwnd_{\max} < cwnd(t) + \beta$, then $cwnd(t) - 1$. The drawback of this algorithm is that the adjustment of the proactive window is not efficient if there are co-existing Reno connections in its path; since it is less aggressive than Reno's policy, which continues to increase window size until packet loss occurs (Hengartner et al. 2000). Therefore, the throughput will be affected:

$$\theta = 0{,}93 \frac{MTU}{RTT\sqrt{\rho}}. \qquad (13)$$

There MTU – maximal transmitted packet size; ρ - probability of the packet losses. To prevent TCP Vegas algorithm from failures, Reno can be modified inasmuch as the decrease of throughput will not be so hard. In order to do this, the best features of the TCP Reno and Vegas algorithms should be combined. These modifications were proposed by Peng Fu and Soung C. Liew (Fu and Liew 2003). The new algorithm was named TCP Veno. Its operation is described by the following mathematical expressions:

$$cwnd(t + t_A) = \begin{cases} cwnd(t) + 1, \text{if } N < \beta; \\ cwnd(t) + \dfrac{1}{cwnd(t)}, \text{if } N \ge \beta, \end{cases} \qquad (14)$$

$$ssth(t + t_A) = \begin{cases} \dfrac{4cwnd}{5}, \text{if } N < \beta; \\ \dfrac{cwnd(t)}{2}, \text{if } N \ge \beta. \end{cases} \qquad (15)$$

Notwithstanding what type of the packet losses are in the link TCP Veno estimates connection state. Using TCP Veno in the typical wireless networks with the random packet loss rate of 1%, the throughput improves to 80% (Fu and Liew 2003). The main feature of Veno is that it modifies only the sender-side protocol of Reno without changing the receiver-side protocol stack.

LONG RANDOM DELAY INFLUENCE TO THE TCP ALGORITHMS

To avoid the random packet losses in the wireless networks reliable MAC layer protocols can be used (Karl et al. 2004). The most important feature of these protocols lies in the repeated packet sending until the ACK packet of MAC layer is received (Airhook 2005). Therefore, the delay of higher OSI protocols increase. In the TCP case the *RTT* time increase. This is affecting $cwnd(t)$ (Peng et al. 2001). The delay of reliable protocols is random, therefore only the random TCP packets will be delayed.

When TCP Reno is used, short delay will determine short *RTO* time (2), therefore the next-transmitted packet with the long *RTT* can be lost (RFC 2581). On the other hand, packet with long *RTT* will increase *RTO* time (2), therefore the random packet loss can be delayed (RFC 793). Furthermore, long *RTO* period can result in the slow-start phase. Thus, it can be proposed that the throughput of TCP Reno depends from random long delayed packet count percentage.

When Vegas and Veno are used $RTT >> RTT_{\min}$ thus N (11) increase, and consequently $cwnd(t)$ will decrease. TCP session capacity will decrease as well (Hengartner et al. 2000).

The analysis of the IEEE 802.11b networks using the reliable "TurboCell" protocol and located at different places in Lithuania has showed that dominant $\overline{RTT_{\min}}$ time is 128 ms, while the $\overline{RTT_{\max}}$ time range is 1.2-2 s, when the average of packet loss approximately is 1 %. It was detected that in some cases $RTT_{\max}$ can reach up to 5 s. The measurements were provided by "Karlnet Configurator" application (Karl et al. 2004).

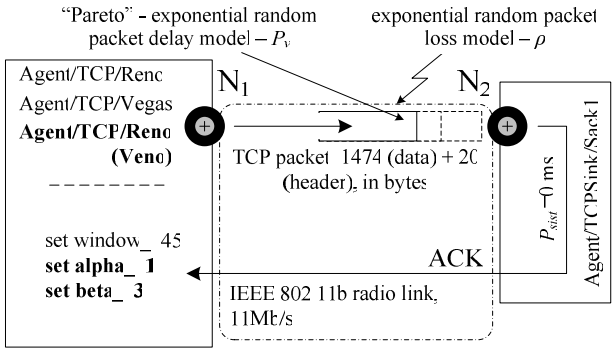
THE METHODOLOGY OF MODELING

To find the impacts on random packet losses and random long delay influence on TCP algorithms used in the IEEE 802.11b networks the modeling of TCP protocol algorithm was performed. For that purpose the "NS-2" modeling engine was used (Fall 2003).

During the research three way of modeling where implemented. While doing this, the information was transmitted between two IEEE802.11b nodes (Fig. 3.). Reno, Vegas and Veno algorithms were used. For the generation of information the FTP flow model with data segments of 1494 bytes was used (Fall 2003).

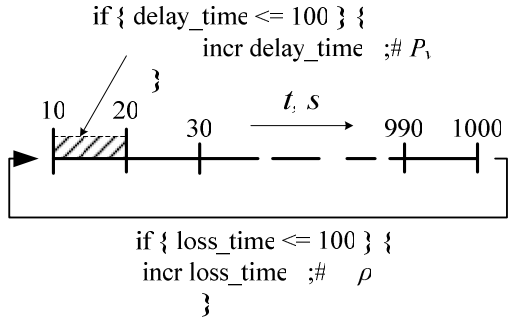
Researching TCP algorithms the congestion window of size $45 \cdot MTU$ and TCP system delay of 0 ms were set. In the Vegas and Veno case $\beta = 3$ and $\alpha = 1$

coefficients were fixed (Fig. 2.). The 11 Mbps physical throughput was defined. Radio link and the exponential random loss models were used between the nodes "Free Space" (Fall 2003).



Figures 3: Modeling Scheme of TCP Algorithms

During the modeling the random long delay percentage rate - P_v (Fig. 4.) was changed 100 times every 10 s. The probability of random packet loss - p was changed every 1000 s. The duration of each modeling was 3 h (Fig. 4.).

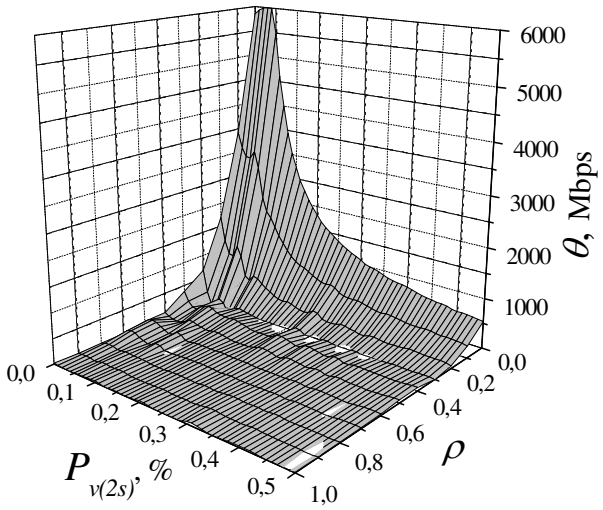


Figures 4: Modeling Temporal Chart

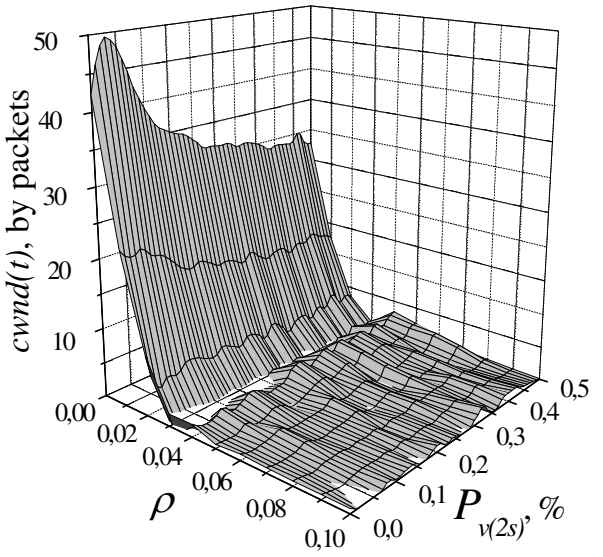
MODELING OF TCP RENO ALGORITHM

The performance of TCP Reno modeling in the IEEE 802.11b network environment has showed that the throughput depends on the probability of the random packet loss and on the percentage of long delay packets. When the probability of random loss did not exceed 10^{-2} the throughput decreased by 40 % (by 3 Mbps). The minimum ($\theta \approx 10kbps$) is reached when the probability of random packet loss is $6 \cdot 10^{-2}$ (Fig. 5.). Under these conditions $cwnd(t)$ usually shifts to slow-start phase (3) (4). Therefore, congestion window becomes minimal (Fig. 6.). If the probability of packet loss is more than $6 \cdot 10^{-2}$ the TCP moves to settle estimated *RTO* duration expectation. Thus, the connection becomes impossible. The results obtained from modeling correspond to the real experimental researches (Fu and Liew 2003).

When transmission TCP Reno is used the throughput of the 2 s long random delay of 10 % packets decrease by 50 % (by 3 Mbps). Since the change of TCP Reno $cwnd(t)$ does not depend on *RTT* duration (1) $cwnd(t)$ depends on delay characteristics (if random packages will be delayed for a long duration). In the modeling the exponential law was used.



Figures 5: Bias Throughput of TCP Reno against Random Packet Loss and Random Long Delay



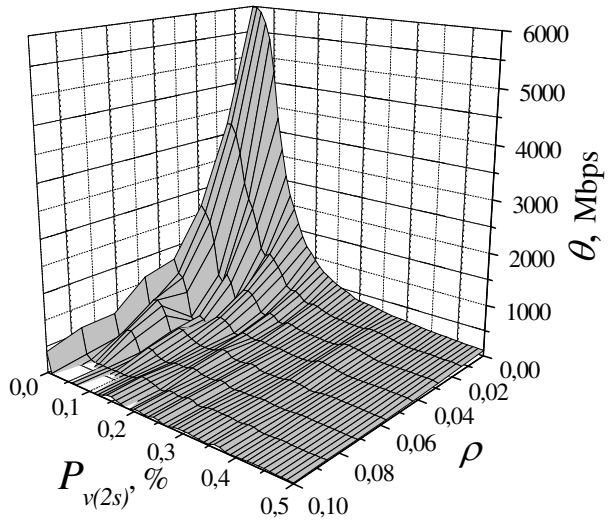
Figures 6: Bias Congestion Window Size of TCP Reno against Random Packet Loss and Random Long Delay

MODELING OF TCP VEGAS AND VENO ALGORITHMS

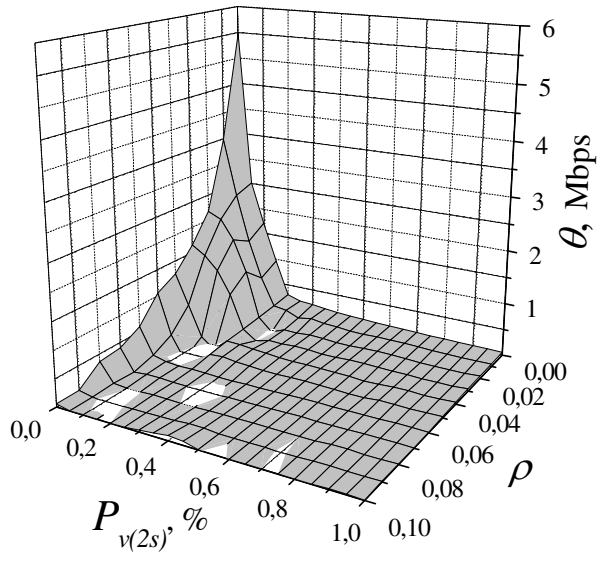
The results obtained from the modeling of TCP Veno algorithm has revealed that under the circumstances of 10^{-2} random packet loss the throughput decrease by 40 % (by 3 Mbps) (Fig. 7.) (Like with TCP Reno). However, under the circumstances of $3 \cdot 10^{-2}$ random packet loss the throughput has increased by 40 % (1.4 Mbps) (Fu and Liew 2003) in comparison to Reno (0.8 Mbps).

The dependence of throughput decrease, if compared to TCP Reno, is smaller when the information is transmitted by Veno algorithm in 2 s random delay. During 2 s random delay of 10 % packets the throughput

differs quite slightly – just 2.2 Mbps. Nevertheless, when the percentage increases up to 30 % the throughput of TCP Reno becomes larger by 65 % (1.5 Mbps). That could be explained by the fact that TCP Veno algorithm is modified by TCP Reno, according to the operating principle of TCP Vegas.



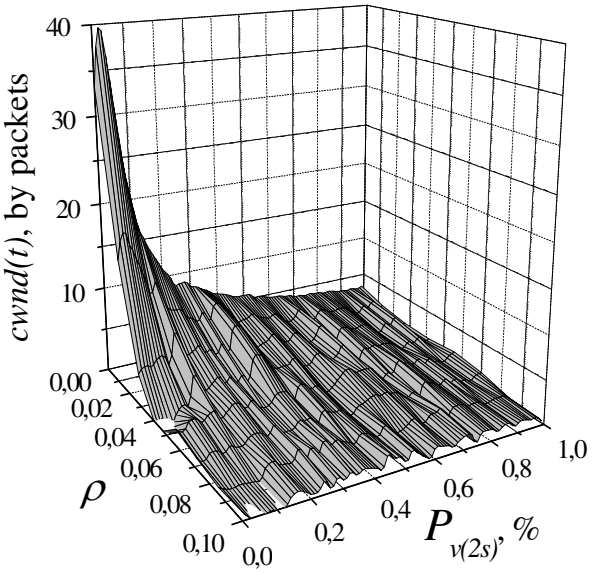
Figures 7: Bias Throughput of TCP Veno against Random Packet Loss and Random Long Delay



Figures 8: Bias Throughput of TCP Vegas against Random Packet Loss and Random Long Delay

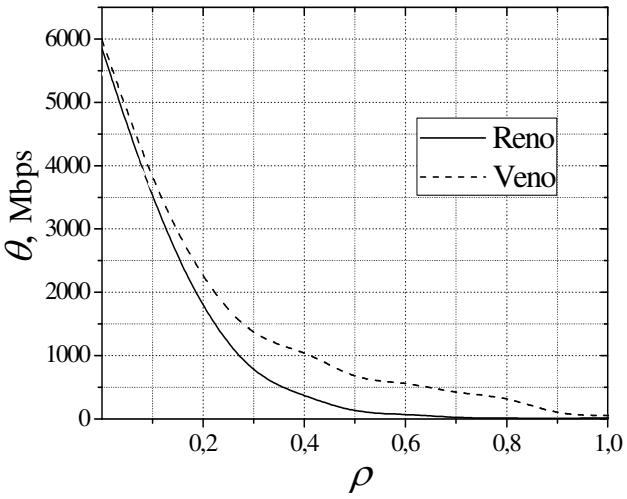
In the case of TCP Vegas, the throughput dependence on random packet loss coincides with TCP Veno characteristics. Since $cwnd(t)$ of the algorithm depends on *RTT* time (12) the considerable change of *RTT* time will determine the constant alternation of $cwnd(t)$ (Brakmo et al. 1994). Therefore, if the random delayed packet percentage is 20 % or even more, the throughput decrease to minimum.

When TCP Veno algorithm is used the characteristics of throughput depends on the size of congestion window (Fig. 9.) proportionally. The characteristics of $cwnd(t)$ depends on random packet loss and long delay characteristics used in mentioned modeling (14) (15).



Figures 9: Congestion Window of TCP Veno against Random Packet Loss and Random Long Delay

The results obtained during the modeling of the TCP Reno and Veno with 0.5 and 1 s delays show that in each analyzed case the throughput of TCP Veno is lower in comparison to Reno (Fig. 10.).



Figures 10: Bias Throughput of TCP Reno and Veno against Random 0.5, 1, and 2 s Delays

CONCLUSIONS

During the modeling of TCP Reno, Vegas, and Veno algorithms in the IEEE 802.11 link environment the results of bias throughput against random packet loss and random long delay were obtained. Under the base of these results the following inferences should be drawn:

1. Using TCP Reno algorithm the connection is possible only when the random packet loss is not higher than $6 \cdot 10^{-2}$. When random packet loss is 10^{-2} the throughput decreases by 40 %. If the percentage of long

delayed packet is increasing the average size of congestion window exponentially decreases, because of the deviation of *RTT* time.

2. TCP Vegas is not adapted to high deviation of delay, therefore when the random delayed packet percentage count is 20 % or more, the throughput decrease even by 95 %.

3. Using TCP Veno under the 10^{-2} random packet loss, the throughput corresponds with Reno under the same conditions. Nevertheless, when the random packet loss reach $3 \cdot 10^{-2}$ the throughput increase by 40% (1.4 Mbps). While 2 s random delay of packages determines worse dependence on throughput in comparison with TCP Reno. If the percentage count of random delay packages increases to 30 % the throughput becomes less by 65 %.

4. Using TCP Veno under the percentage count of 2 s delay the throughput corresponds with Reno under the same conditions. Nevertheless, when percentages count of delayed packages reach 30 % and more the throughput increases by 75 % (1 Mbps).

In conclusion, it may be assumed that random packet loss and random long delay affect the throughput when any of the analyzed TCP algorithms are used. If random packet losses are very frequent the TCP Veno algorithm should be used, and if random long delay becomes long the TCP Reno should be used in the wireless IEEE 802.11b networks environment.

REFERENCES

Airhook – Reliable Data Delivery Protocol. 2005. http://airhook.org/protocol.html

Balakrishnan H.; Padmanabhan V. N.; Seshan S.; and Katz R. H. 1997. "A Comparison of Mechanisms for Improving TCP Performance Over Wireless Links". *IEEE/ACM. – Transactions on Networking* 5, Vol. 6, 756-769.

Brakmo L. S.; O'Malley S. W.; and Peterson L. L. 1994. "TCP Vegas: New Techniques for Congestion Detection and Avoidance". *SIGCOMM*. 24-35.

Buyn H. J. and Lim J. T. 2005. "On Affair Congestion Control Scheme for TCP Vegas". *IEEE Communications Letters*, Vol. 9, No. 2

Fall K. 2003. *The Ns Manual.* UC Berkeley, LBL, USC/ISI, and Xerox PARC, http://www.isi.edu/nsnam/ns.

Floyd S. and Jacobson V. 1993. "Random Early Detection Gateways for Congestion Avoidance". *IEEE/ACM. – Transactions on Networking* 4, Vol. 1, 397-413.

Fu C. P. and Liew S. C. 2003. "TCP Veno: TCP Enhancement for Transmission Over Wireless Access". *IEEE. – Communications 21*, Vol.2.

Ghani N. and Dixit S. 1999. "TCP/IP Enhancements for Satellite Networks**".** *IEEE. – Communications Magazine*, Vol. 37, 64-72.

Hengartner U.; Bolliger J.; and Gross T. 2000. "TCP Vegas Revisited". In *INFOCOM'2000* Conference, Vol. 2, 1546-1555.

Jacobson V. and Karels M. J. 1988. "Congestion Avoidance and Control". University of California at Berkeley.

Jeonghoon M.; Richard L. L; Venkat A.; and Walrand J. 1999. "Analysis and Comparison of TCP Reno and Vegas". *IEEE/ACM. – Transactions on networking*. Vol. 6, 1556-1563.

Karl D.; Monahan S.; and Whitman A. 2004. "KarlNet's TurboCell: Enhancing the Capabilities of Standard 802.11". www.karlnet.com.

Kumar A. 1998. "Comparative Performance Analysis of Versions of TCP In a Local Network With a Lossy Link". *IEEE/ACM. – Transactions on networking*, Vol. 6, Nr. 4.

Lakshman T. V. and Madhow U. 1997. "The Performance of TCP/IP for Networks with High Bandwidth-delay Products and Random Loss". *IEEE/ACM – Transactions on Networking*, Vol. 5, No 3 (July), 336-350.

Mo J.; La R. J.; Anantharam V.; and Walrand J. J. 1999. "Analysis and Comparison of TCP Reno and Vegas". In *INFOCOM'99* Conference, Vol. 3, 1556-1563.

Partridge C. and Shepard T. J. 1997. "TCP/IP Performance Over Satellite Links". *IEEE/ACM. – Transactions on networking*, No. 9 (Sept.), 44-49.

Peng J.; Andreadis P.; and Barbeau C. B. M. 2001. "Improving TCP Performance over Long Delay Satellite Links" In *OPNETWORK'2001* Conference, Washington, Vol. 1.

Saltis A. and Pavilanskas L. 2003. "Paketinio radijo rysio 2,4 GHz dazniu ruoze eksperimentiniai tyrimai". *Elektronika ir elektrotechnika*, Kaunas, Technologija, No. 46, 48-56.

LUKAS PAVILANSKAS was born in 1979, in Taurage, Lithuania. He received his M.S. degree in Electronics science from Vilnius Gediminas Technical University in 2003. He is currently doctoral student at telecommunications engineering department at the Vilnius Gediminas technical university. He also is an IEEE Student member of Communications Society. Basic research area – modeling of wireless telecommunications systems (NS-2, OpNet), QoS in IEEE 802.11 networks, and reliable wireless communications. Ongoing research projects: Adaptation of MAC protocol to voice communications in the wireless packet networks. He has participated in 7 conferences and has 3 published works. His e-mail address is: lukas.pavilanskas@el.vtu.lt and his Web-page can be found at http://tc.el.vtu.lt.