TCP with Bandwidth Estimation over Wireless Networks

Fabio Martignon, Antonio Capone
Politecnico di Milano, Dipartimento di Elettronica e Informazione, Italy
{martignon,capone}@elet.polimi.it

Abstract— **The Time Intervals based Bandwidth Estimation Technique (TIBET) is a proposed change to the TCP congestion control algorithm which performs a run-time sender-side-only estimation of the bandwidth available to the connection. TIBET uses the estimated bandwidth to allow TCP sources to achieve higher performances both over wired and wireless networks, avoiding overly conservative rate reductions in presence of random losses typical of wireless links.**

This paper analyzes the performances of TIBET sources over wireless links by accurately modeling correlated losses due to fading or handover typical of cellular networks. An analytical model developed in this paper allows to quantify the advantage of TCP sources performing an explicit bandwidth estimation, as TIBET, over actually implemented TCP versions, i.e TCP Reno and NewReno.

Keywords - TCP, Bandwidth Estimation, Fairness, Wireless Links, Performance Model.

I. INTRODUCTION

The Transmission Control Protocol (TCP) has shown to be efficient in classical wired networks, proving its ability to adapt to modern high-speed networks and new scenarios for which it wasn't originally designed. However, the extraordinary success of modern wireless access networks, including cellular networks and wireless links, poses new challenges to the TCP congestion control scheme. The actual versions of TCP, such as Reno and NewReno, experience heavy throughput degradation over channels with high error rate, such as wireless links. It is known [1] that the main reason to these degradated performances is the confusion the classical TCP versions make between random losses, due for example to sudden fading over wireless channels, and true congestion signals due to high network load. A possible approach to this problem is to modify the TCP congestion control scheme by implementing an explicit estimation of the bandwidth used by the connection. Such algorithms have been recently proposed in the literature for TCP Vegas and TCP Westwood [2]. They both perform an explicit bandwidth estimate and use this estimate to avoid unnecessary reductions of their transmission rate by trying to distinguish between random losses and true congestion.

However, we have shown in [3] that all these schemes do not perform well, either because the estimation scheme is highly biased (TCP Westwood), or because new TCP sources are unable to fairly share network resources (TCP Vegas). For these reasons, we have proposed a new scheme, the Time Intervals based Bandwidth Estimation Technique (TIBET) which succeedes in obtaining good estimates of the bandwidth used by a TCP source. TIBET can be implemented by modifying the sender-side only of a TCP connection, and this implies the possibility of its immediate deployment all over the Internet.

In this paper we'll point out the benefits of the proposed scheme compared to existing versions of TCP in networks with wireless links affected by random losses, either correlated and uncorrelated, typical of wireless environments.

Finally, as a good estimate is a necessary condition for every algorithm performing a sender-side estimate, we'll briefly examine the behavior of an ideal algorithm having always a perfect estimate of the fair-share bandwidth. We will show, however, that even in this case there's a physical limit which cannot be easily overcome by using this sender-side only estimate approach, and we'll provide an empirical model which explaines why this happens.

The rest of this paper is structured as follows: in Section 2 we briefly illustrate the TIBET algorithm. In Section 3 we show how TIBET can fairly share network resources with existing TCP Reno sources. In Section 4 we analyze TIBET performances in presence of wireless links, affected either by independent and correlated losses. A semi-empirical model is proposed in Section 5 for a TIBET scheme performing an ideal estimate, thus allowing to discover the upper bound in the throughput which can be obtained over lossy links by TCP sources which perform an explicit bandwidth estimate. Finally, Section 6 concludes the paper.

II. TIBET ALGORITHM

In this section we present the Time Intervals based Bandwidth Estimation Technique (TIBET), which succeedes to obtain correct estimates of the bandwidth used by the TCP source. TIBET is able to cope efficiently with many of the bandwith estimation problems known in literature, including Clustering and ACK Compression [4], [5].

To explain the rationale of TIBET let us refer to the example in Figure 1 where transmissions occurring in a period T are considered. Let n be the number of packets belonging to a connection and $L_1, L_2...L_n$ the lengths, in bits, of these packets.

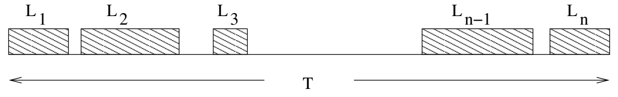

Fig. 1. Packet timing structure

The average bandwidth used by the connection is simply given by $\frac{1}{T}\sum_{i=1}^{n} L_i$. If we define $\overline{L} = \frac{1}{n}\sum_{i=1}^{n} L_i$, we can express the bandwidth (Bwe) used by the connection as:

$$Bwe = \frac{n\overline{L}}{T} = \frac{\overline{L}}{\frac{T}{n}} \tag{1}$$

The basic idea is to perform a run-time sender-side estimate of the average packet length, $\overline{L}$, and the average interarrival, $\frac{T}{n}$, separately. This can be done in two ways:

- by measuring and low-pass filtering the length of acked packets and the intervals between ACKs' arrivals;
- by measuring and low-pass filtering directly the packets' length and the intervals between sending times, according to the self-clocking nature of TCP[6].

The pseudo-code of the bandwidth estimation scheme based on transmitted packets is the following:

```
if (Packet is sent)
length[k]     = 8 * packet_size;
interval[k]   = now - last_sending_time;
Av_length[k]  = alpha * Av_length[k-1]
              + (1-alpha)*length[k];
Av_interv[k]  = alpha * Av_interv[k-1]
              + (1-alpha )* interval[k];
Bwe[k] = Av_length[k] / Av_interv[k]
end if
```

where *length* indicates the segment's size in bits, while *packet_size* indicates its size in bytes. The variable *interval* contains the interdeparture samples, as *now* indicates the current time and *last_sending_time* the time of the previous packet transmission. *k* and *k-1* indicate the current and previous values of the variables. *Av_length* and *av_interv* are the low-pass filtered measures of the packet length and the interdeparture times, respectively. $alpha$ is the pole of the two low-pass filters, with $0 \leq alpha \leq 1$. Finally, *Bwe* is the estimated value of the used bandwidth. This is the algorithm used to obtain the simulation results reported in this paper.

Evidently the value used for $alpha$ presents a trade-off: if $alpha$ has a low value, the proposed algorithm is highly responsive to changes in the available bandwidth, but the oscillations of $Bwe[k]$ are quite consistent; on the contrary, if $alpha$ approaches 1, TIBET produces more stable estimates, but it's less responsive to network changes. After having considered several network scenarios, we have found that with $alpha = 0.99$ TIBET achieves a good compromise between the two behaviors described above.

If we consider the minimum RTT measured by the TCP source (RTT_{min}) as a good estimator of the end-to-end propagation delay, then we can set:

$$Ssthresh = Bwe * RTT_{min} \tag{2}$$

which has been proposed as an optimal value for $ssthresh$ in [7].

The *ssthresh* is set to the value of equation (2) only after the receipt of three duplicate ACK's, or after a coarse-grained timeout expiration. More frequent updates to the *ssthresh* value may lead both to worst estimates and overall performance degradation, as we demonstrated in [3]. The congestion window, instead, is set to the same value of $ssthresh$ after three duplicate ACK's, and it's reset to 1 segment after timeout expirations to let the network recover from congestion periods.

Simulation results presented in [3] show that TIBET is not biased, and obtains bandwidth estimates very close to the fair-share value when all TCP sources experience almost the same path conditions.

We said that TIBET algorithm uses the estimated bandwidth only after congestion episodes. This choice is supported by the worst performance observed with more frequent updating of *ssthresh* as discussed in [3].

III. Fairness

TIBET can actually achieve an accurate estimate of the used bandwidth. Now we have to check if this allows TCP sources using TIBET to share network resources fairly with other existing versions of the protocol.

Figures 2 and 3 show the results of fairness tests referring to a mixed scenario with TIBET and TCP Reno connections. The scenario considers a total of 10 connections sharing a 2 or a 10 Mbit/s bottleneck link, respectively. These results and all the others presented in this paper were obtained using Network Simulator (NS) [8], for which we developed TIBET modules. The x-axis of the two Figures represents the number of TCP Reno connections used in the simulation, the remaining connections use TIBET. The y-axis value represents the average throughput obtained by the two types of algorithms, and the dotted line is the fair-share. We observe that the throughput achieved by both algorithms is very close to the fair share.

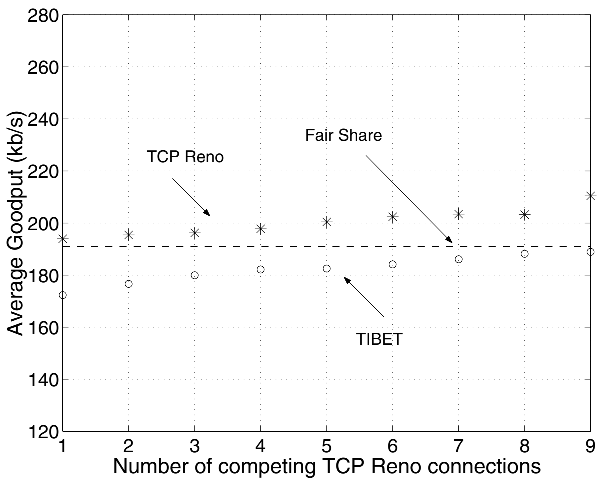

Fig. 2. TIBET fairness towards TCP Reno over a 2 Mbit/s link

We have also run simulations over different scenarios covering link bandwidths ranging from few kbit/s to 150 Mbit/s, varying the number of competing connections and using also a more complex topology with multiple congested gateways. The results obtained are almost the same: TIBET obtains the same level of fairness as TCP Reno. Simulation results also show that TIBET improves its performance when the number of connections sharing the bottleneck link increases since the

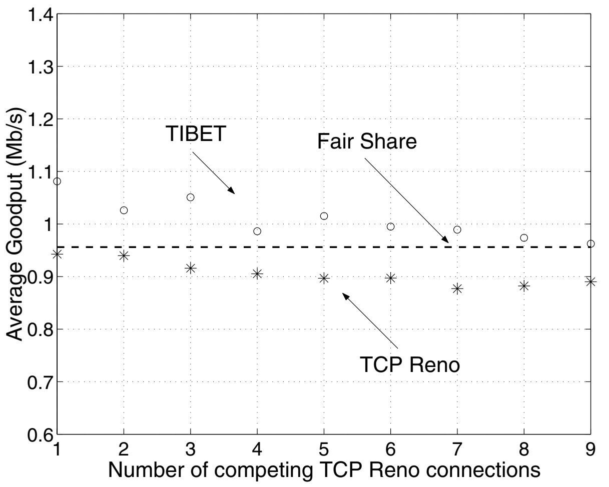

Fig. 3. TIBET fairness towards TCP Reno over a 10 Mbit/s link

estimate variance reduces. Moreover, the presence of constant rate flows, such as UDP flows for IP telephony or video conference, makes TIBET perform better as it reduces the size of packet clusters.

IV. Wireless Links

TIBET performs an efficient bandwidth estimation and fairly shares network resources with other TCP sources. The main goal for which this algorithm has been designed is to achieve higher performances over links affected by sporadic losses.

A. *Uncorrelated Losses*

To this purpose, in Figures 4 and 5 we compare the throughput achieved by a single connection running TIBET to that of a TCP Reno connection over links affected by random errors. The link has a capacity of 2 Mb/s in Figure 4 and of 10 Mb/s in Figure 5, and in both cases the FIFO DropTail queue can store a number of packets equal to the bandwidth-delay product. The one-way RTT is 50 ms, and the link drops packets according to a Poisson process with average ranging from 0.01% to 10%.

We observe that TIBET can sustain higher throughput than TCP Reno at all drop rates considered. This is due to the filtering process which keeps in account also the past history of the bandwidth estimates avoiding to confuse network congestion signals due to queue drops with losses due to link errors. Similar results have been obtained with various link speeds.

Packet losses are not the unique cause of TCP throughput degradation. Many studies proposed in the literature [1], [9] have pointed out that TCP performance also degrades when the round trip time of the connection increases. TIBET allows TCP sources to alleviate this degradation and to obtain better performance. To prove this, we have compared in Figure 6 the goodput achieved by a single TCP Reno and TIBET source transmitting over a 10Mb/s link affected by a packet loss probability constantly equal to 0.1%. The delay of the link is increased in each simulation in order to vary the RTT of the connections from 20ms to 400ms.

Evidently, TIBET obtaines higher goodput than TCP Reno with every RTT value.

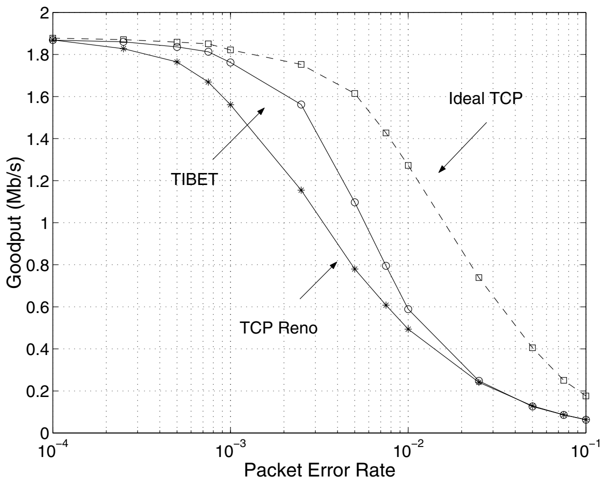

Fig. 4. TIBET and TCP Reno goodput vs link error rate over a 2Mb/s link

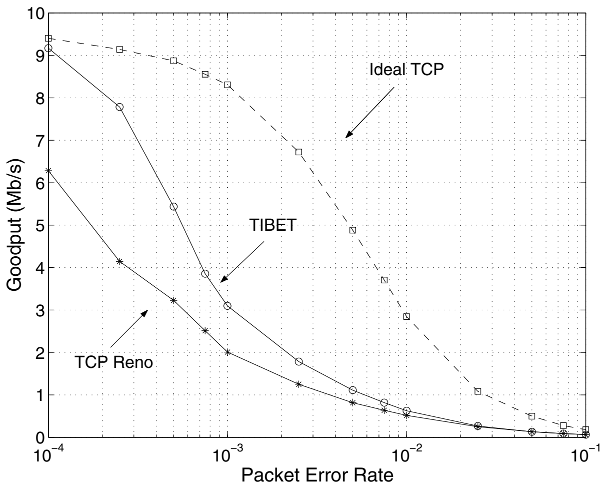

Fig. 5. TIBET and TCP Reno goodput vs link error rate over a 10Mb/s link

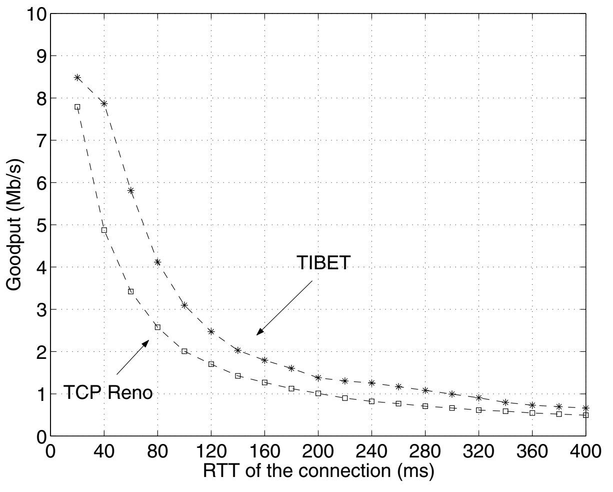

Fig. 6. TIBET and TCP Reno goodput over a 10Mb/s link with increasing RTT and packet loss probability constantly equal to 0.1%

B. *Correlated Losses*

TIBET has an edge of advantage over TCP Reno also over links affected by correlated losses. Wireless links are typically affected by correlated losses due to a variety of situations associated mainly to the mobility of terminals as cellular phones. These situations include fading and blackout caused, for example, by handoffs.

According to existent literature, we have modeled the wireless link state (*Good* or *Bad*) with a two-state Markov chain, as illustrated in Figure 7. The duration of the two states is exponentially distributed with parameters λ_g for the *Good* state and λ_b for the *Bad* one.

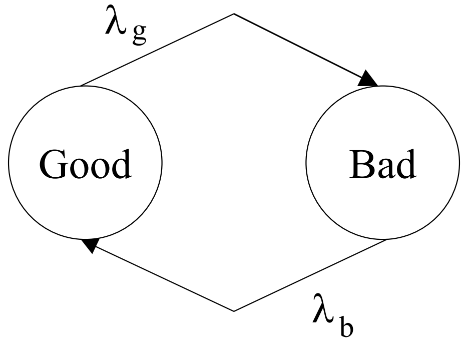

Fig. 7. Markovian two-state link model

We have considered a wireless link capacity equal to 2 Mb/s, and average durations of *Good* and *Bad* states equal to 8 and 4 seconds, respectively. The packet loss probability in the *Good* state is constant and equal to 1%, while we varied the one in the *Bad* state from 0 to 50% to take into account various levels of fading. Figure 2 shows that, even in this case, TIBET algorithm obtains higher goodput than TCP Reno.

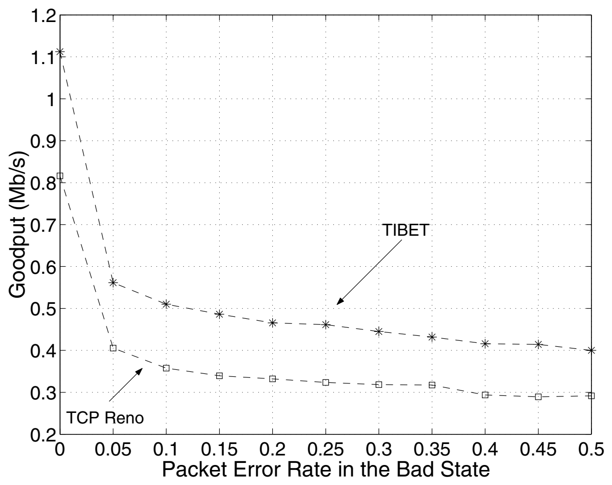

Fig. 8. TIBET and TCP Reno goodput as a function of the packet error rate in the Bad state over a 2 Mb/s link affected by correlated losses

V. Performance Model

It is useful to find out how far we are from the throughput upper bound that all schemes exploiting the new bandwidth estimation scheme cannot overcome. To this purpose we have considered a TCP which has a perfect knowledge of the exact bandwidth available along the path in every instant (Ideal TCP) and uses this value to set the *ssthresh* accordingly. Its performance has been reported in Figures 4 and 5.

Evidently, the throughput achieved by this ideal scheme which is higher than that obtained by TIBET. This is due to the estimation errors but also to fact that TIBET can only estimate the used bandwidth and not the available one. However, we also observe that the throughput collapse at high error rates is nevertheless experienced by Ideal TCP. Hence, this prove that this behavior does not depend by the estimation algorithm performed by the TCP source.

Several analytical models have been proposed in the literature for the most commonly implemented TCP versions as Reno. Detayled expressions for TCP Reno throughput have been developed in [9], [1], [10], and we'll use these results to provide a comparison between Reno and Ideal TCP.

To evaluate throughput upper-bounds of TCP algorithms performing direct bandwidth estimations we have developed a simple semi-empirical model, which has been validated by simulation results. Here we only report the final approximate expression of the goodput achieved by Ideal TCP sources, which has been obtained dividing the average number of bits sent between two loss events and the average duration of the period itself:

$$Goodput_{Ideal} \propto \frac{MSS}{RTT}\frac{1}{p \cdot log_2(\frac{1}{p})} \qquad (3)$$

where p represents the packet error rate, RTT is the Round Trip Time and MSS the Maximum Segment Size of the connection. This approximation is valid within the range of error rates typical of wireless networks, i.e. for p ranging between 10^{-3} and 10^{-1} as shown in [11]. This result is particularly interesting, expecially if compared to the goodput achieved by TCP Reno connections running over lossy links, which has been shown in literature [9], [1] to be proportional to

$$Goodput_{Reno} \propto \frac{MSS}{RTT}\frac{1}{\sqrt{p}} \qquad (4)$$

Figure 9 compares the asymptotic behaviors of the goodput achieved by the ideal algorithm (using expression 4) and by TCP Reno. In both case, we used the correct coefficient of proportionality, i.e. 1.31 ([1]) and 3.4 for TCP Reno and for TCP performing an ideal estimate, respectively. For more details about the complete performance model and the approximation please refer to [12] Figure 9 can be seen as a comparison between the goodput achieved by the two different algorithms when both RTT and MSS are equal to 1.

In order to validate our model with simulation results we considered a simple scenario (Figure 10) with a wired link between the source S and the Base Station BS, and a wireless one, with packet error rate equal to p, between BS and the destination D.

Figures 11 and 12 show the numerical results obtained with connections having a RTT of 100ms and 400ms, respectively. We observe that our simplified model is able to predict quite well the throughput achieved by the TCP connections and it is very accurate when the packet error rate is high.

VI. Conclusions

In this paper we have discussed the issues related to the use of enhanced bandwidth estimation algorithms for TCP congestion

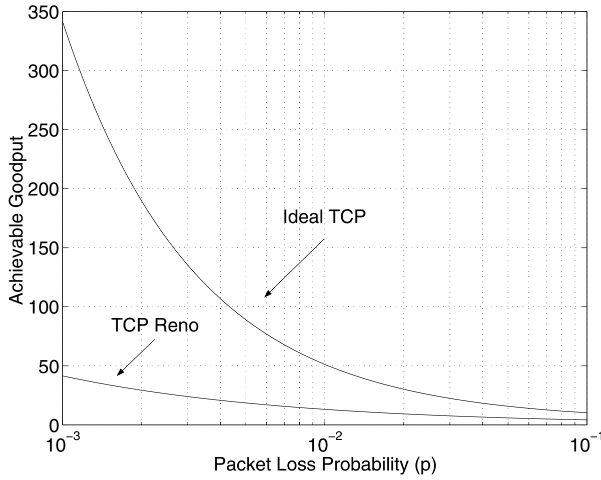

Fig. 9. Approximate performance comparison between TCP Reno and TCP with an Ideal Estimate

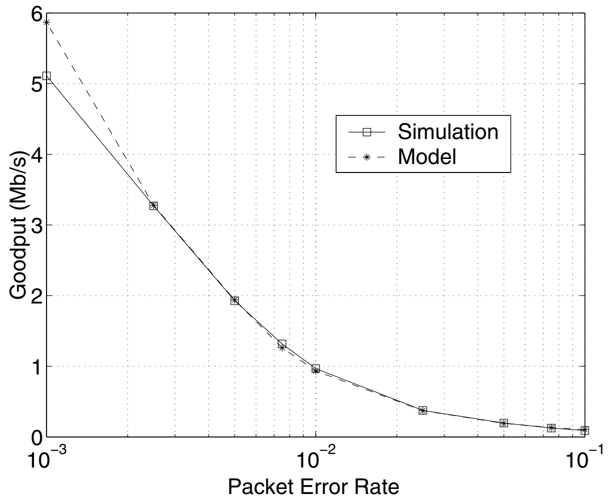

Fig. 12. Comparison of analytical model and simulation results with RTT = 400ms

control. These algorithms can potentially improve the throughput of TCP over wireless links affected by independent or correlated errors. However, previously proposed schemes performing bandwidth estimation do not fairly share resources with TCP Reno over wired links and thus they cannot be smoothly introduced in the Internet. For this reason we have proposed TIBET, a new algorithm, which allows TCP to achieve a higher throughput over wireless links while keeping a good fairness level over wired links. We have shown that the key issue for fairness is estimation accuracy and we have developed an approximate model which considers an ideal scheme with an error free bandwidth estimation. This model provides an upper bound on the throughput of all schemes based on the new bandwidth estimation approach.

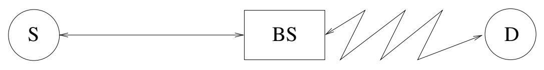

Fig. 10. Simulation environment for model validation

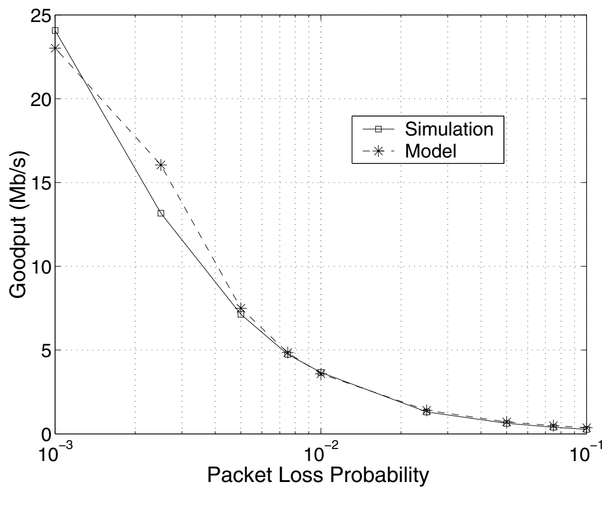

Fig. 11. Comparison of analytical model and simulation results with RTT = 100ms

REFERENCES

[1] M.Mathis, J.Semke, and J.Mahdavi. The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. *ACM Computer Communications Review*, 27(3), 1997.

[2] S.Mascolo, C.Casetti, M.Gerla, M.Y.Sanadidi, and R.Wang. TCP Westwood : End-to-End Bandwidth Estimation for Efficient Transport over Wired and Wireless Networks. In *Proceedings of ACM MOBICOM'01*, 2001.

[3] A.Capone and F.Martignon. Bandwidth Estimates in the TCP Congestion Control Scheme. In *Proceedings of Tyrrhenian International Workshop on Digital Communications (IWDC 2001)*, Taormina, Italy, Sep.2001.

[4] L.Zhang, S.Shenker, and D.Clark. Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic. In *Proceedings of SIGCOMM'91 Symposium on Communications Architectures and Protocols*, pages 133–147, Zurich, September 1991.

[5] J.C.Mogul. Observing TCP Dynamics in Real Networks. In *Proceedings of ACM SIGCOMM'92 Symposium on Communications Architectures and Protocols*, pages 305–317.

[6] V.Jacobson. Congestion avoidance and control. *ACM Computer Communications Review*, 18(4):314–329, 1988.

[7] J.C.Hoe. Improving the Start-up Behavior of a Congestion Control Scheme for TCP. *ACM SIGCOMM Computer Communications Review*, 26(4):270–280, October 1996.

[8] ns-2 network simulator (ver.2).LBL. URL: http://www.isi.edu/nsnam.

[9] J.Padhye, V.Firoiu, D.Towsley, and J.Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. In *Proceedings of ACM SIGCOMM '98*, 1998.

[10] F.Kelly. Mathematical modeling of the Internet. In *Proceedings of the Fourth International Congress on Industrial and Applied Mathematics*, pages 105–116, 1999.

[11] G.Xylomenos, G.C.Polyzos, P.Mahonen, and M.Saaranen. TCP Performance Issues over Wireless Links. *IEEE Communications Magazine*, 39(4):52–58, April 2001.

[12] A. Capone and F. Martignon. Enhanced Bandwidth Estimation Schemes for TCP Congestion Control. Technical report, Politecnico di Milano Technical Report #2002-11, 2002.