

TRANSPORT CONTROL PROTOCOLS FOR WIRELESS CONNECTIONS

Hala A. ElAarag and Mostafa A. Bassiouni

School of Computer Science

University of Central Florida

Orlando, FL 32816

e-mail: elaarag@cs.ucf.edu bassi@cs.ucf.edu

Abstract- In this paper, we study the effect of bit error rates and handoff disconnections on the performance of different TCP implementations (Tahoe, Reno, New-Reno, and SACK). Test scenarios with larger link-up periods, when the mobile is connected, but higher disconnection probability yielded better performance than scenarios with smaller link-up periods and smaller disconnection probability. Our performance tests show that the performance of TCP is more sensitive to the length of the link-up period than the error rate of the wireless link, or the mobile disconnection probability. While SACK TCP is well known to perform better than Reno TCP in wired networks, the tests show that this is not the case in mobile networks. The paper presents performance results for the four TCP implementations in a mobile wireless environment and provides comparisons based on important performance metrics (throughput, goodput, transfer time) and multimedia QoS measures (average packet delay).

I. INTRODUCTION

Transport Control Protocol (TCP) [1], [2] is widely used in network applications because of its reliable transmission. The emerging mobile networks will allow users to access data anywhere using the wireless infrastructure. However, the performance of TCP has been well tuned for traditional networks made up of wired links and stationary hosts. Mobile networks, therefore, raise a new set of technical problems, as they are not subject to the location constraints of the wireline infrastructure plus the lossy nature of the wireless link. Nevertheless, mobile users will expect the same QoS they get on a wired network.

TCP was designed to relate lost packets to congestion. However, in mobile wireless networks packets are usually lost for reasons other than congestion. The wireless links suffer from high bit error rates (BER). Recent studies [3] indicate that BER may be no worse than 10^{-5} . Forward Error Correction codes (FEC) and retransmission schemes at the link layer may reduce the

BER by one (possibly two) orders of magnitude [4]. While this can impact the performance of TCP, other factors like mobile disconnections are more important than BER. In this paper, we study the effect of the disconnection probability and the length of the link-up period on TCP performance. We also compare the performance of four different implementations of TCP: Tahoe TCP [5], Reno TCP [6] [7], New-Reno TCP [8] and Sack TCP [9], in the presence of wireless link BER and frequent mobile disconnection.

The rest of the paper is organized as follows. Section II describes our simulation model, TCP parameters and the performance metrics we used. Section III and IV describe the effect of the mobile disconnection probability and the link-up period, respectively, on the TCP performance. Section V presents the comparison between four different implementations of TCP against our performance measures. Finally, section VI presents the conclusion of this paper.

II. SIMULATION MODEL

We chose a network topology that consists of three nodes: a fixed host (FH), a base station (BS) and a mobile host (MH). This topology is motivated by the many recent experiments of TCP performance over wireless mobile links; see for example [10]. There is a wired link between the fixed host and the mobile host of bandwidth 1.5Mb and delay (D1) 10ms. There is a wireless link between the base station and the mobile host of bandwidth 0.8Mb and delay (D2) 100ms. There is a DropTail queue for the base station and the mobile host. We have a TCP connection between FH and MH where FH is the source and MH is a sink. BS merely transfers data received from FH to MH. Bulk data are transferred via ftp from FH to MH. The wireless link is lossy and has bit error rate of BER. Its error model is characterized by two states: a good state and a bad state. In the good state, the wireless link does not suffer any losses, while, in the bad state the packets get corrupted. Both states are exponentially distributed with means α_g and α_b , respectively. The simulation model

has been implemented using the Network Simulator (NS) developed by Lawrence Berkeley Labs. MH has an exponential disconnection probability model. The link-up state, where MH is connected is exponentially distributed with mean link-up time (m_1). The link-down state, where MH is disconnected is also exponentially distributed with mean link-down (m_2). MH has a disconnection probability p , where $p = m_2 / (m_1 + m_2)$.

TCP Parameters

A window size is set to 50, with ssthresh of 32. The packet size is chosen to be 1024 bytes, while ACKs are set to 40 bytes. The TCP clock granularity is set to 100ms, this means that times are calculated to the nearest 100ms. The queue sizes are set to 50 to avoid the drop of packets due to buffering. We focus only on the loss of packets due to link error rate and mobile disconnections.

Performance Metrics

We used four performance metrics: throughput, goodput, delay and transfer time. Throughput is an indication of how much data the user can receive per second. Goodput is determined as the ratio between the amount of useful data transmitted by the source and the total amount of data transmitted by the source. We define packet delay to be the average delay incurred by the packet from the time it is deposited into the sender's window until it is successfully acknowledged. Notice that the packet delay used in this paper includes the queuing delay to wait for transmission within the window (thus larger windows tend to produce larger delay). Transfer time is determined by the time it takes the receiver to receive a fixed number of packets, say 5000 packets. This is an indication of how fast the user can receive the requested data. Better quality of service for the user is obtained with higher throughput, smaller average packet delay and smaller transfer time.

III. EFFECT OF MOBILE DISCONNECTION PROBABILITY

Experiment: We ran experiments on four different implementations of TCP: Tahoe, Reno, New-Reno and Sack. For each implementation we had two scenarios: scenario1 had a disconnection probability $p = 9\%$ (link-up period = 1.0s, link-down period = 0.1s), and scenario2 had a disconnection probability $p = 28.6\%$ (link-up period = 5.0s, link-down period = 2.0s), which is much larger than the first scenario.

Observation: Table 1 shows the results for Tahoe TCP. Simulations are run for 1000 seconds.

	Scenario1 ($p=9\%$)	Scenario2 ($p=28.6\%$)
Throughput	5.275	26.671
Goodput	0.8768	0.96
Transfer time for 5000 pkts	138.734	16.4604

Table 1. Throughput, Goodput, and Transfer time for Tahoe TCP with $p=9\%$ and $p=28.6\%$

The disconnection probability in scenario2 is more than three times as that of scenario1. Nevertheless, its throughput is about 5 times and its goodput is about 1.1 times that of scenario1; its transfer time is about 8.4 times less than that of scenario1.

Based on the above, the performance of the network does not only depend on the disconnection probability but more on the length of the mean of the link-up period. Scenario2 had a larger link-up mean, and that is why it resulted in a better performance, even though it had an unrealistically high disconnection probability. We notice that the reason for the bad performance of TCP in scenario 1 is that the sender does not have enough time to open up a large window and send packets to MH before it disconnects. Although it disconnects for a very short time, by the time the sender starts sending again, the mobile disconnects again, and so on. The short connection time (1 sec) and the frequent disconnections made FH unable to send enough packets to MH. On the other hand, in scenario2, there is a long connection time for FH to send packets before the mobile disconnects. Even though, the mobile disconnects for a long period (2sec), but the frequency of the disconnection in this scenario was much less than the first.

IV. EFFECT OF LINK UP PERIOD

For a disconnection probability of 9%, we varied the mean of the link-up period and found that the larger the mean of the link-up period the better the throughput, the goodput and the transfer time. The performance of TCP with a disconnection probability of the mobile of 9% can be better than that with a disconnection probability of 28.6% only with a suitable mean of link-up period. We repeated the experiments for Tahoe, New-Reno and SACK TCP and they provided similar results. Figure 1 shows that for the same disconnection probability, the higher the mean of the linkup, the larger the throughput of TCP. The figure also shows the throughput of TCP with disconnection probability $p=28.6\%$ ($m_1 = 5$, $m_2 = 2$) results in a throughput of 26.67, which is better than the throughput with $p=9\%$ and m_1 is less than 3.

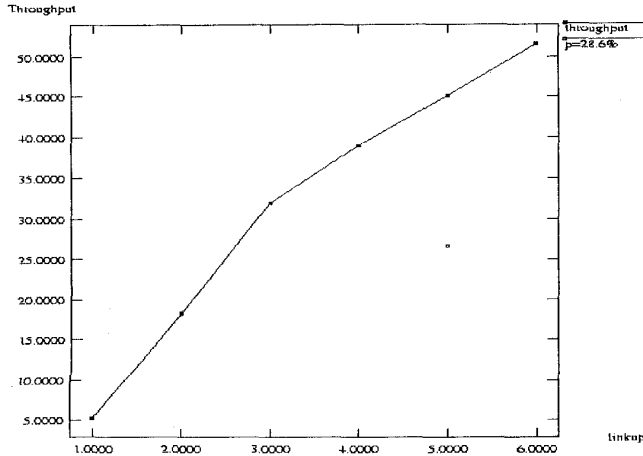


Figure 1. Throughput of Reno TCP vs m1 for p=9%

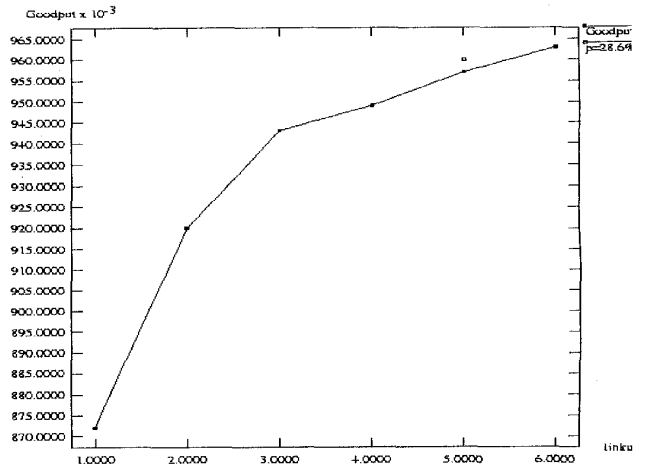


Figure 2. Goodput of Reno TCP vs m1 for p=9%

Figure 2 shows for p=9%, the higher m1 is, the higher the goodput of the network. The goodput with p=28.6% (m1=5; m2=2) results in goodput of 0.96, which is better than the goodput with p=9% and m1 is less than 5. This means that the goodput of the network is highly affected by the value of m1. The utilization of the network is always better with higher values of m1. In Figure 3, we show the effect of m1 on the transfer time for sending 5000 packets. The higher m1, the less the transfer time. From figure 3 one can notice the value of the transfer time becomes constant (52.29s) for m1 greater or equal to 5. In this case, the TCP connection does not suffer any disconnection while sending the 5000 packets. For p=28.6% (m1=5, m2=2) the transfer time was also 52.29s. This means that the transfer time is highly sensitive and dependent on the value of m1.

We noticed from our simulations, that the disconnection of the mobile has a greater impact on the performance of TCP than the effect of BER. Disconnections, generally can be of the order of several seconds. They can even last for 1 minute [11]. This result in the loss of a large number of contiguous data packets and ACKs, causing frequent timeout of the TCP connection and resulting in a degradation of its performance. Combining this observation with the results of the effect of the link-up period, we can conclude that the link-up period has the dominant effect on the performance of TCP.

In the next section, we will compare the behavior of the different implementations in a wireless network.

V. PERFORMANCE OF TAHOE, RENO, NEW-RENO AND SACK TCP IN WIRELESS NETWORKS

We ran simulations to compare the performance of Tahoe, Reno, New-Reno and Sack TCP in a wireless mobile network, considering both mobile disconnections and bit error rates.

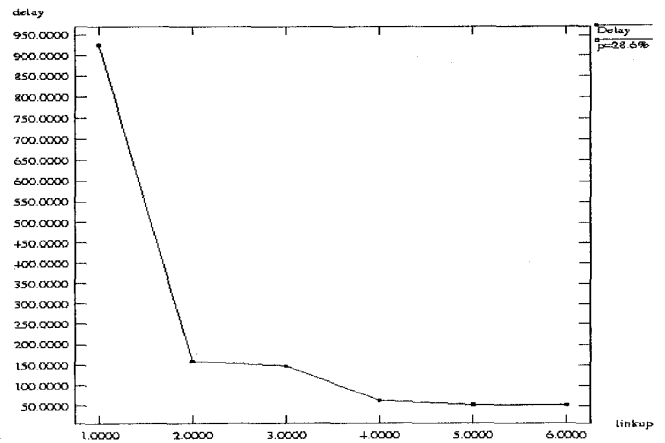


Figure 3. Transfer time of Reno TCP vs m1 for p=9%

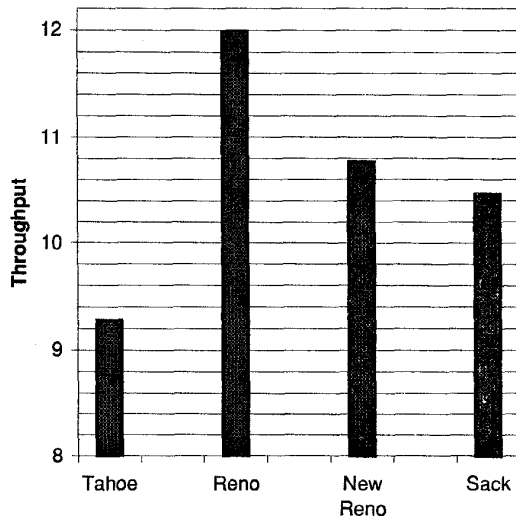


Fig. 4. Throughput of TCP Implementations

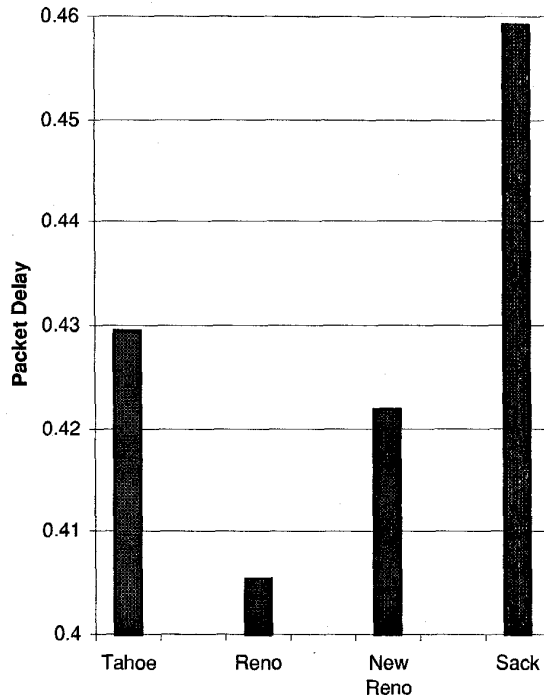


Fig. 5. Packet delay of TCP Implementations

We chose the lossy wireless link to have BER of 10^{-5} . Thus, for TCP packets (BS to MH) $\alpha_g = 100$, $\alpha_b = 8$ while for ACKs (MH to BS) $\alpha_g = 1000$, $\alpha_b = 3$. For the mobile disconnectivity, $m1=2.0$ and $m2=0.3$. Performance results are presented in figures 4-6. The performance of the network in the No Loss case (ideal case where the network does not suffer any link errors or mobile disconnections) is given in Table 2. The

figures for delay and transfer time in this table are in seconds; notice that the delay in the No Loss case is large because it includes the queuing time within large windows (as discussed earlier).

Throughput	97.549
Goodput	1
Transfer time for 5000 packets	52.2902
Average packet Delay	0.511596

Table 2. Performance Measures in an ideal network

Figures 4 and 5 show the throughput and average packet delay respectively, for the four considered TCP implementations over a period of 600 seconds. Figure 6 shows the transfer time versus the number of packets. Figures 4-6 show that Reno had the highest throughput, the least average packet delay and the least transfer time. In terms of throughput and the least transfer time, the implementations were ordered as Reno followed by New-Reno, then Sack, and finally Tahoe i.e. the more the throughput of the implementation, the less its transfer time.

Figure 7 shows a plot of the congestion window (cwnd) for Tahoe TCP. The values of cwnd are taken from a log of runtime of 1800 seconds. The plot is the zoom of the cwnd vs. time in the interval of 1200 seconds and 1300 seconds, for illustration. When comparing the plots, including cwnd plots for Reno, New-Reno and Sack (not included here for space limitation), we notice that the Tahoe plot has the largest number of occurrences of cwnd=1 values. This is because Tahoe TCP slow-starts every time a packet is dropped. This caused Tahoe to have the least average cwnd (9.4015) and the least performance, refer to figures 4-6. We compared the plots to study the differences between the four implementations. We also collected the following statistics. We found out that Reno TCP was the most sensitive protocol in changing the values of cwnd, i.e. for the same time period, 20299 values of cwnd were logged, compared to 15555, 18602 and 18252 for Tahoe, New-Reno and Sack respectively. This means the more a protocol changed the value of cwnd, the better it was adaptive to the need of the network, the better its performance was. Nevertheless, Reno did not have the largest average cwnd (9.87) compared to 9.97 and 10.42 for New-Reno and Sack, respectively. Reno was able to timeout 588 times, compared to 572 and 567 for New-Reno and Sack, respectively. This seems to be a good feature of Reno and the reason for its better performance compared to the other protocols. This may mean that the more the protocol was able to timeout the better its performance in wireless networks, and not the larger the window it was able to have. In the case of a wireless network with high frequency of

disconnections, as the scenario of our simulation, multiple contiguous packets and acknowledgements are dropped every time the mobile disconnects. Thus the only way to recover is to timeout. This is because in this case there is no way the source will receive duplicate acknowledgements. Now we can conclude that, maintaining a large open window is important in wireless networks, but the more important is to fast adapt to the requirement of the network. Reno had better performance than New-Reno and Sack because it was able to timeout more, on the other hand, it was better than Tahoe because of its fast recovery algorithm.

VI. CONCLUSION

In this paper, we studied the effect of mobile disconnection probability and the mobile link up period on the performance of TCP in a wireless mobile network. We showed that the mobile link up period has the dominant effect on the performance of TCP than the disconnection probability of the mobile and even the BER of the wireless link. We compared the behavior of Tahoe, Reno, New-Reno and Sack in the presence of wireless link errors and mobile disconnections. We used four important performance metrics in our comparison: throughput, goodput, transfer time and average packet delay. We showed that Reno had the most throughput, least transfer time and least average packet delay.

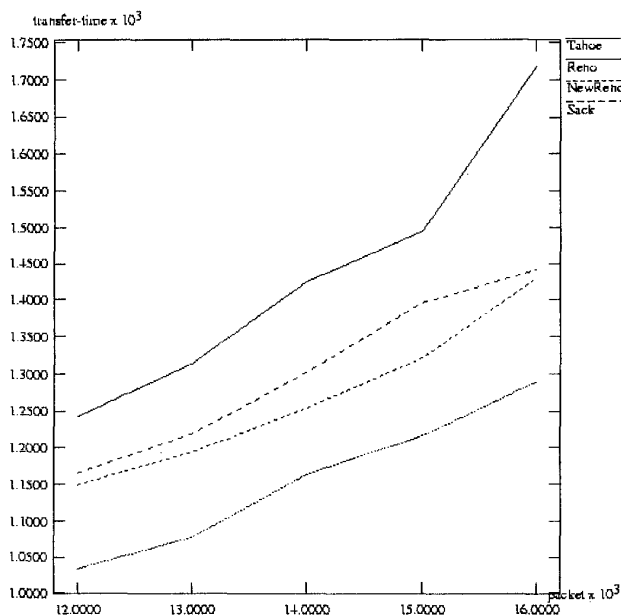


Figure 6: Transfer time vs. # of packets of TCP implementations

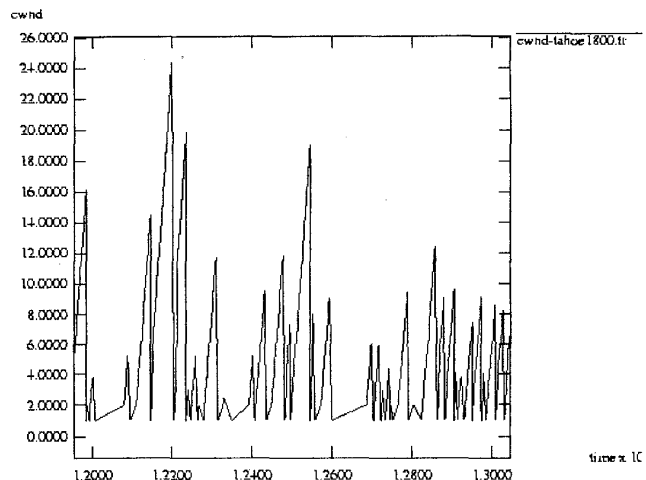


Figure 7: Congestion window for Tahoe TCP

Acknowledgment

This work has been supported by ARO under grant no. DAAH04-95-1-0250 and by an I-4 corridor grant from the State of Florida and Harris Corporation.

REFERENCES

- [1] Comer D., *Internetworking With TCP/IP*, Vol.1, 2 &3, Prentice Hall, 1991.
- [2] Stevens W.R., *TCP/IP Illustrated*, vol. 1, Addison-Wesley, 1994.
- [3] Eckhardt D., and Steenkiste P., "Measurement and Analysis of the Error Characteristics of an In-Building Wireless Network", *Proceedings ACM SIGCOMM*, pp.243-254, October 1996.
- [4] Ayanoglu E., et al., "AIRMAIL: A Link-Layer Protocol for Wireless networks", *Journal of Wireless Networks*, Vol. 1, pp. 47-60, 1995.
- [5] Jacobson V. "Congestion Avoidance and Control" *ACM SIGCOMM'98*, pp. 314-329, 1998.
- [6] Jacobson V. "Modified TCP Congestion Avoidance Algorithm" Tech Report 30, 1990.
- [7] Stevens W.R. "TCP Slow Start, Congestion Avoidance, Fast Retransmission, and Fast Recovery Algorithms", *IETF*, RFC 2001, 1997.
- [8] Hoe J. C., "Improving the Start-up Behavior of a Congestion Control Scheme for TCP", *ACM SIGCOMM 1996*, pp.270-280.
- [9] Mathis M., Mahdavi J., Floyd S., and Romanow A., "TCP Selective Acknowledgment Options", *IETF*, RFC 2018, October 1996.
- [10] Bakshi B. et al., "Improving Performance of TCP over Wireless Networks", *17th International Conference on Distributed Computing Systems*, pp. 365-373, 1997.
- [11] Brown, K., and Singh S., "M-TCP: TCP for Mobile Cellular Networks", *ACM SIGCOMM Computer Communication Review*, pp. 19-43, July 1997.