An enhanced TCP mechanism – Fast-TCP in IP networks with wireless links

Jian Ma [a], Jussi Ruutu [b] and Jing Wu [c]

[a] *Nokia China R&D Center, No. 10, He Ping Li Dong Jie, Beijing 100013, P.R. China*
[b] *Nokia Research Center, P.O. Box 422, FIN-00045 NOKIA GROUP, Finland*
[c] *National Key Lab of Switching Technology and Telecommunication Networks, Beijing University of Posts & Telecommunications, P.O. Box 206, Beijing 100876, P.R. China*

This paper presents the results of the first study on performance of Fast-TCP over wireless links with transmission errors. We have studied a case in which a TCP source is connected with a TCP receiver over two routers. The link between the routers is wireless and has bit errors. We compare the performance of Fast-TCP with plain TCP in the wireless environment. We evaluate Fast-TCP using the OPNET Radio Modeler^{TM} discrete event simulation tool. Simulation results show that even without any adaptations of other wireless protocols Fast-TCP still has more advantages over plain TCP.

1. Introduction

Transmission Control Protocol [2], TCP, is the most widely used transport layer protocol in the Internet. The recent years have witnessed a widespread use of this protocol that dates back to the beginning of 1980s. Obviously, the origins of TCP are in fixed networks.

Wireless data connections are becoming widely deployed with the introduction of wireless LANs and cellular networks. One of the motivations for third generation cellular networks has been the provision of higher data rates than those of the current second generation networks. A rapidly increasing number of TCP connections will be carried over wireless links at some parts of the path, as shown in figure 1. To provide easy interaction between hosts in the existing Internet and mobile hosts in wireless networks, TCP should be used in mobile devices as effectively as possible.

However, networks with wireless links incur significant losses due to bit-errors and handoffs. This environment violates many of the assumptions made by TCP, causing degraded end-to-end performance.

For example, the flow control of TCP assumes that all packet losses are due to the network congestion [5]. As a consequence, TCP enters to the slow start phase as a response to packet loss. In practice, the performance of TCP decreases rapidly even though there is no real network congestion.

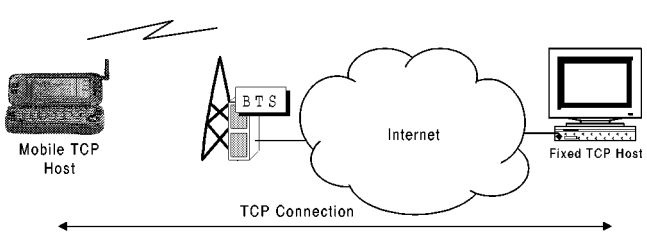

Figure 1. TCP connection over a wireless link.

There exist several methods that can be used for optimizing the performance of TCP. In this paper, we have studied the performance of Fast-TCP method [3] over wireless links that have bit errors.

2. Wireless links and TCP

Wireless links differ in many ways from fixed links. First of all, wireless links typically have much higher bit error rates. This means that either TCP segments can be corrupted (unreliable links) or their transmission time over a reliable link may vary a lot. In the former case there is no link layer retransmission while in the latter case the link layer takes care of the retransmissions over the link.

A corrupted TCP segment will cause the expiration of TCP retransmission timer. However, even in the case of a reliable link, the transmission time over the wireless path may vary so that the TCP retransmission timer expires. In both cases the TCP enters to slow start phase and the transmission window drops to one segment. This occurs even if the real problem is not network congestion but the performance of the link layer.

Another potential problem is caused by handovers (or handoffs) that are typical in many wireless networks. During the handover it is likely that some TCP segments are delayed or even lost. For example, some segments buffered in a base station (see figure 1) may be forwarded to another base station, if the mobile station decides to make a handover before the segments are transmitted over the air interface.

Thus, there is a clear demand for methods that can suppress the problems caused by wireless environment.

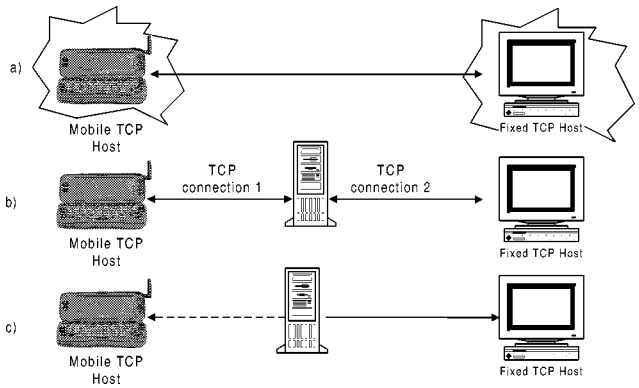

Figure 2. TCP optimisation method categories. (a) Modification of end hosts. (b) Split TCP, an intermediate node terminates the TCP connection. (c) Snoop TCP, the TCP connection is modified at an intermediate node.

3. TCP optimisation

Several techniques have been developed to improve end-to-end TCP performance over wireless links. They can be roughly classified into three categories: (1) end-to-end TCP, (2) split TCP and (3) snoop TCP kind of methods, as shown in figure 2.

The idea of the first category of methods is to make modifications to the end systems, such as Fixed TCP host and Mobile TCP host in figure 2(a). While modifications to the transmitting and receiving TCP protocol implementations can allow very effective improvements, there is a major drawback.

Namely, it is not reasonable to expect that all the existing TCP implementations are modified to support better wireless performance. Moreover, the TCP end hosts may not even know that there is a wireless link in between.

The second approach, split TCP, is based on the idea of terminating the TCP connection before the wireless link, for example, at the base station (BTS) in figure 2(b). A separate TCP connection is then used over the wireless link between the base station and the mobile host. This approach does not necessarily require any changes in the end hosts but certainly breaks the end-to-end semantics associated with the TCP protocol.

The third category of TCP optimization methods is based on the idea that the flow of TCP segments is intercepted in the middle of the connection and manipulated. For example, an intermediate router in figure 2(c) may acknowledge TCP segments and take care of retransmissions so that the fixed TCP host does not see the wireless link in its flow control [1].

4. Fast-TCP

4.1. General

A TCP acknowledgement (ACK) delay proposal, called Fast-TCP [3], has recently been proposed to be used for more accurate control of the TCP transmission rate. The basic idea of the method is that a network element, such as a router, delays the IP packets carrying TCP acknowledgements, when network congestion is about to occur. Since the TCP source does not receive an acknowledgement, it keeps its current transmission window until the delayed acknowledgement is received.

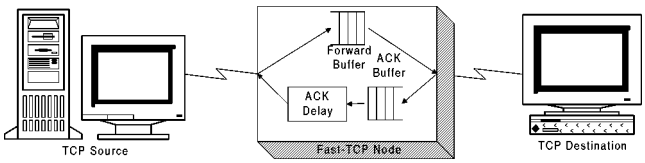

Figure 3. Fast-TCP.

It has been proved that Fast-TCP can speed up TCP flow control time, reduce buffer oscillation, increase bandwidth utilization, increase throughput, and reduce packet losses in the IP networks with wired links [6]. From implementation point of view only about few tenths of a buffer in a normal router are needed to achieve the same performance as plain TCP in some scenario environments. In addition, Fast-TCP is implemented at the intermediate router, which eliminates the need to modify the TCP stacks of the source or destination.

4.2. Operation of Fast-TCP

The basic idea of Fast-TCP method is to delay the ACKs being transferred from the TCP destination towards the TCP source (see figure 3). This can be done, for example, at the base station where congestion has been detected. The Fast-TCP flow control mechanism is located on the output of the access unit to the IP interface (in backward connection) and controls ACK output rate according to congestion information from forward connection. Instead of discarding packets on the forward connection, the congested node delays ACKs on the backward connection and thus causes the TCP source to reduce its output rate.

The load of the network is monitored in the Fast-TCP node, for example, by monitoring the buffer occupancy in forward connection, as shown in figure 3. If overload is detected, i.e. if buffer occupancy exceeds a predefined level, a congestion notification is sent inside the node to a delay controller in the backward connection. The delay controller carries out a congestion control algorithm that decides the delay value for TCP acknowledgements. Next ACKs travelling at that moment through the router towards the traffic sources are delayed. In this way the TCP source, which operates in the manner described in [2], automatically starts to slow down its transmission rate, or at least it does not increase its transmission rate as quickly as it otherwise would. This is because the delay slows down the rate at which the source increases the size of its congestion window.

It is also important to note that the transport protocol TCP does not have to be modified in any way since Fast-TCP operates essentially at IP level. Only the IP packets carrying TCP ACKs should be identified so that the correct packets are delayed.

5. Wireless connections with Fast-TCP

This paper presents the results of the first study on performance of Fast-TCP over wireless links with transmission errors. We have studied a case in which a TCP source is connected with a TCP receiver over two routers, as illustrated in figure 4. The link between the routers is wireless. The simulation setup is described in detail in section 5.1. In section 5.2 we present our simulation results when the wireless link does not have any errors. In section 5.3 we describe our results with a link that contains errors.

All the simulation studies presented here have been made using the OPNET Radio Modeler™ discrete event simulation tool.

5.1. Simulation setup

The setup is shown in figure 4. Router 1 contains a Fast-TCP system with a forward buffer for TCP segments carrying data, an ACK buffer for TCP acknowledgements flowing in the backward direction and ACK delay controller, as illustrated in figure 3 for a Fast-TCP node.

ACK delay controller operates using the following principles. Let Q be the forward buffer occupancy, *Th* the congestion notification trigger threshold, D the interval between releasing two consecutive ACKs and d the minimum time difference between two data packets in the most congested link. The delay between released ACKs is determined using the rule:

$$\text{if } Q < Th \text{ then } D = d,$$
$$\text{else } D = 2d.$$

In other words, when the forward buffer occupancy exceeds the threshold, the time between ACKs released from the ACK buffer is doubled.

This is naturally a very simple scenario, and it is possible to produce a Fast-TCP node with more complex algorithms. For example, several thresholds may be applied. However, we selected this straightforward approach since we wanted to test a robust, basic method with wireless links.

The forwarding rate of the TCP source and destination is 1 000 000 packets/s, buffers for both directions are 2 Mbit, TCP initial RTO is 1 s, and TCP maximum segment size is 512 bytes. The routers (1 and 2) have IP forwarding rate of 14 000 packets/s, buffers of 2 Mbit and congestion notification threshold *Th* is 400 000 bits. The links between the routers and TCP hosts is 155.52 Mbit/s while the wireless link between the routers is 30 Mbit/s. The propagation delay from the TCP source to Router 1 and from Router 2 to the TCP destination is 1 ms, while the propagation delay over the wireless link from Router 1 to Router 2 is 130 ms.

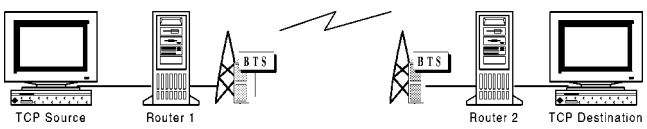

Figure 4. Simulation setup. TCP source and destination are connected over two routers that have a wireless link in between.

The TCP transmission window defines the size of the burst that the TCP source sends. Problems occur if the transmission window grows big enough to overflow a buffer with too big a burst. In our scenario the packet losses can be expected to occur in Router 1, since the wireless link out of Router 1 is five times slower than the link carrying traffic from the TCP source to Router 1.

5.2. Connection over a perfect link

In the first set of simulations we studied the case when the wireless link did not contain any bit errors. Two simulations were performed, the first without Fast-TCP (plain TCP) and the second with Fast-TCP using otherwise the same simulation parameters.

The operation of Fast-TCP method can be seen in the occupancy levels of the forward and backward buffer of Router 1. Figure 5 illustrates the occupancy of the forward buffer of Router 1 in bits. The size of the buffer is 2 Mbit.

The figure shows clearly the bursts caused by the increasing TCP transmission window. In the case of plain TCP, the successive occupancy peaks roughly double in size. This corresponds to the doubling of the TCP transmission window during slow start phase. When the size of the burst overflows the buffer at time $t = 14.4$ s, the buffer becomes full (level hits 2 Mbit). This results in packet losses and decreased performance of TCP.

However, when Fast-TCP is used, at the same time the occupancy stays at 0.4 Mbit, well below congestion level. In practice, the delaying of TCP ACKs spreads the burst so that there is more time to send TCP segments out of Router 1 through the slow wireless link.

Figure 6 illustrates the occupancy of backward buffer. Here, we can see the price of using Fast-TCP. If Fast-TCP is not used, the occupancy stays close to zero. When Fast-TCP is deployed, the buffer occupancy increases rapidly when the algorithm starts to delay ACKs. Notice that the occupancy peaks of backward buffer correspond at time axis to the operation of Fast-TCP seen at figure 5.

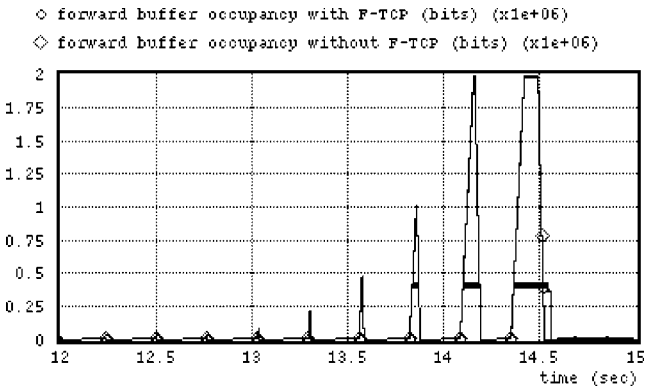

Figure 5. Forward buffer occupancy of Router 1 in bits. The size of the buffer is 2 Mbit. Notice that at about $t = 14.4$ s the buffer becomes full without Fast-TCP.

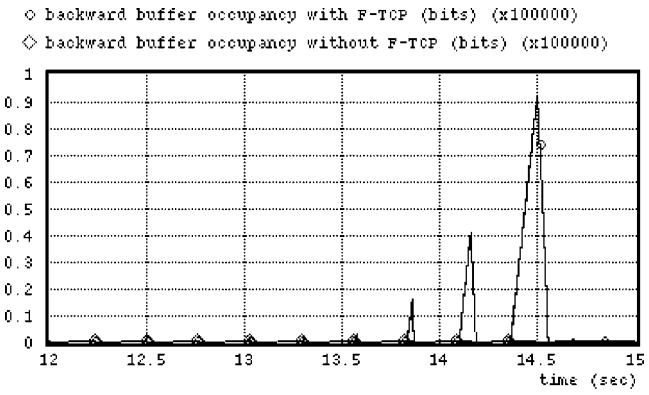

Figure 6. Occupancy of backward buffer of Router 1 in bits.

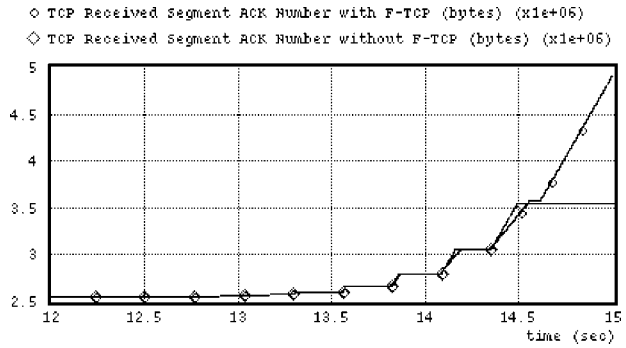

Figure 7. Received TCP segment ACK number at the TCP source. Notice the divergence after $t = 14.5$ s.

However, here one should also remember that the typical size of a backward buffer packet (acknowledgement) is smaller than the size of a forward packet (data packet). Thus, the overall buffer occupancy (forward + backward buffer) in Router 1 can be decreased when using Fast-TCP.

The benefit of Fast-TCP can be seen from figure 7. The figure illustrates the received TCP segment ACK number at the TCP source. It can seen that at $t = 14.4$ s without Fast-TCP the number becomes almost stationary due to the lost TCP segments and the resulting slow start. At the same time a system with Fast-TCP increases rapidly the amount of successfully transmitted data.

5.3. Connection over a link with bit errors

In the second set of simulations we have studied the performance of TCP with links containing bit errors.

The parameters of the simulation were the same as in the previous case, but now the wireless link produced errors with the bit error rate (BER) 10^{-8} errors/bit.

Figure 8 illustrates the received TCP segment ACK number at the TCP source. At time $t = 25$ s, the system with Fast-TCP has been able to transfer TCP segments up to $10.8 \cdot 10^6$ while without Fast-TCP the same figure is $3.5 \cdot 10^6$. Thus, there is a clear improvement when using Fast-TCP.

Actually, this result may appear first surprising since Fast-TCP cannot do anything about the corrupted TCP segment. This is evident, since Fast-TCP is, after all, a flow control mechanism. Thus, here is no difference between the performance of Fast-TCP and plain TCP routers. When a bit error destroys a TCP segment, the TCP source goes to the slow start phase.

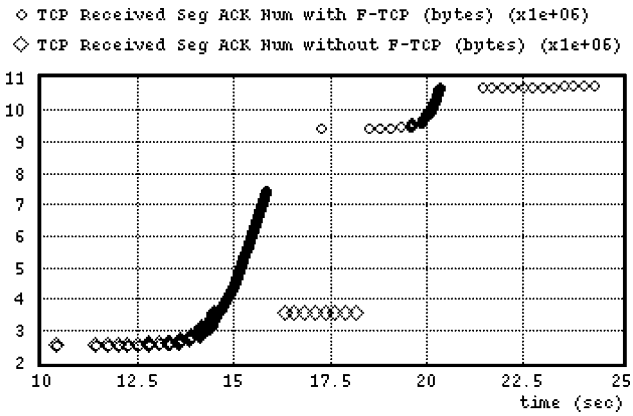

Figure 8. Received TCP segment ACK number at the TCP source. Imperfect links.

However, Fast-TCP starts to show its effect after the packet loss. As shown in the results of section 5.2, Fast-TCP is able to prevent the buffer overflow during slow start when the TCP transmission window increases rapidly. Since the imperfect wireless link causes lost segments and, as a consequence, slow starts, the Fast-TCP has even more opportunities to act. Naturally, the overall amount of data transferred is higher with the perfect link.

We also found out that in some cases even the Fast-TCP method is not sufficient to reduce the effect of corrupted TCP segments. Let us assume that in the beginning of a big burst, a single TCP segment is corrupted. As a result, the TCP receiver receives successfully all other segments except for the corrupted one. If the TCP source successfully retransmits the corrupted segment, the TCP receiver may generate a single acknowledgement that covers the whole burst. Once the TCP source receives this ACK, it will slide the transmission window in a big, single step and generate a rapid, big burst that overflows the forward buffer in Router 1. The basic Fast-TCP cannot do much here since delaying of this single "oversized" ACK does not help.

6. Conclusions

We have studied the performance of Fast-TCP over a wireless link. In the case of a perfect link, the effect of Fast-TCP is to prevent the congestion of the forward buffer in a router in front of the slow wireless link. Our results indicate that Fast-TCP improves significantly the performance of TCP.

In the second set of simulations we have studied the performance of Fast-TCP over a wireless link with errors. Naturally, Fast-TCP cannot do anything about the corrupted packets. However, Fast-TCP shows its effectiveness after

the packet loss, during the slow start phase by reducing burstiness.

The main conclusion is that, even without any special adaptations to wireless protocols, Fast-TCP improves the performance of wireless TCP connections. In our simulation model, the only input for Fast-TCP method has been the occupancy of the forward buffer. Thus, no wireless specific information has been utilized. On this ground, the studied Fast-TCP method appears a robust, simple and straightforward way to improve the performance of TCP connections.

Acknowledgements

This article is based on our previously published material from wmATM'99 workshop, organized by Wireless Mobile ATM Task Force of Delson Group.

References

[1] H. Balakrishnan, S. Seshan and R. Katz, Improving reliable transport and handoff performance in cellular wireless networks, in: *Proceedings of the First ACM Conference on Mobile Computing and Networking (Mobicom '95)*, Berkeley (November 1995).

[2] IETF RFC793, Transmission Control Protocol (1981).

[3] J. Ma, Interworking between TCP and ATM flow controls, ATM Forum/97-0960.

[4] J. Ma, A simple fast flow control for TCP/IP over satellite ATM network, in: *wmATM '98*, Hangzhou, China (6–10 April 1998).

[5] W.R. Stevens, *TCP/IP Illustrated*, Vol. 1: *The Protocols* (Addison-Wesley, Reading, MA, 1994).

[6] J. Wu, P. Zhang, T. Du, J. Ma and S. Cheng, Improving TCP performance in ATM network by the fast TCP flow control, in: *ICCT '98*, Beijing, China (22–24 October 1998).

Jian Ma. Biography not available at time of publication.
E-mail: Jian.janne.ma@ntc.nokia.com

Jussi Ruutu. Photograph and biography not available at time of publication.
E-mail: jussi.ruutu@research.nokia.com

Jing Wu. Photograph and biography not available at time of publication.
E-mail: wujing@bupt.edu.cn