Improving TCP Performance in Mobile Networks

Wanjiun Liao, Chang-Jung Kao, and Chin-Hei Chien

***Abstract*—In this letter, a new transport layer mechanism is proposed to improve the performance of transport control protocol (TCP) in mobile networks. The proposed mechanism is comprised of two parts: a loss classifier (LC) and a congestion window extrapolator (CWE). Based on LC, the cause of packet loss during roaming is determined. If the loss is considered to be caused by congestion in the wireline, the congestion window is halved; otherwise, the packet is considered to be lost in the last hop, the wireless portion, and the sender adjusts the size of the congestion window based on CWE. We conduct simulations to evaluate the performance of the proposed mechanism. The results show that our mechanism significantly improves TCP performance as compared with existing solutions for mobile networks.**

***Index Terms*—Mobile networks, wireless transport control protocol (TCP).**

I. Introduction

TRANSPORT control protocol (TCP) is a transport-layer protocol providing reliable and ordered data service in the Internet. While TCP performs well in wired networks, when used in mobile wireless networks, TCP performance may degrade due to high bit-error rates on the wireless link or temporary disconnections caused by handoffs. Much research effort has been expended to enable wireless or mobile TCP. Existing work on this subject can be classified into two categories: 1) approaches based on base station (BS) assistance, such as I-TCP [1], MTCP [2], Snoop [3], M-TCP [4], and WTCP [5]; and 2) end-to-end approaches, such as fast retransmission [6], explicit bad-state notification [7], and freeze TCP [8].

In this letter, we focus on the performance problem with temporary disconnections caused by handoffs, and propose an end-to-end mechanism to improve the performance of TCP for roaming users in mobile networks. Such disconnections may further result in data loss during the handoff period, and throughput degradation due to handoffs. To solve these problems, the proposed mechanism is comprised of two parts, a loss classifier (LC) and a congestion window extrapolator (CWE). The LC targets the first issue and determines the reason of a packet loss during the handoff period; the CWE focuses on the second issue and improves the throughput of the connection after handoffs. Note that most of the existing work focuses on the enhancement of TCP performance over wireless networks at the receiver side, i.e., the operation is performed at the receiver side. In this letter, we discuss this problem from the perspective of the sender.

Paper approved by Y. Fang, the Editor for Wireless Networks of the IEEE Communications Society. Manuscript received June 14, 2003; revised December 25, 2003; March 3, 2004; November 1, 2004. This work was supported in part by the National Science Council under a Center Excellence Grant NSC93-2752-E-002-006-PAE, and in part by the National Science Council, Taiwan, under Grant NSC93-2213-E-002-132. This paper was presented in part at the IEEE Vehicular Technology Conference, Spring 2002.

The authors are with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. (e-mail: wjliao@ntu.edu.tw).

Digital Object Identifier 10.1109/TCOMM.2005.844921

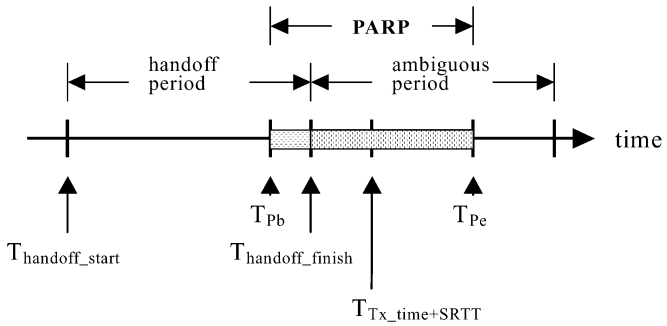

Fig. 1. Illustration of PARP.

The rest of the paper is organized as follows. Section II describes the proposed mechanism. Section III shows the simulation results. Finally, the paper is concluded in Section IV.

II. Proposed Mechanism

In this section, the proposed mechanism, comprised of an LC and a CWE, is described. Our mechanism is implemented at the mobile sender only. The receiver still runs TCP Reno.

A. Loss Classifier (LC)

The LC is used to determine why packets are not received after handoffs. The loss of a packet during a handoff may be caused by congestion in the wireline, or by handoffs, or by transmission errors on the channel. In LC, transmission errors on the channels are treated in the same way as errors due to network congestion. In both cases, the congestion window of the TCP sender is shrunk to half of that of the normal TCP operation.

To determine the cause of a packet loss, two loss probabilities are calculated, P_H and P_C. P_H is the probability that a loss is caused by the handoff, and P_C, the loss is due to congestion. If $P_C > P_H$, the mechanism regards the loss as due to congestion, and the normal congestion control mechanism is initiated; otherwise, the loss is due to handoffs, and the sender can continue sending unsent packets in the usable window size based on the CWE, which will be described later.

We briefly describe how P_H and P_C are calculated. When a timeout occurs, a period called possible acknowledgment returning period (PARP) is calculated according to

$$\begin{aligned} T_{Pb} &= \text{the beginning time of the PARP} \\ &= \text{Transmit\_time} + \text{SRTT} - \text{RTTVAR} \qquad (1) \\ T_{Pe} &= \text{the end time of the PARP} \\ &= \text{Transmit\_time} + \text{SRTT} + \text{RTTVAR}. \qquad (2) \end{aligned}$$

Here SRTT means smoothed Round-Trip Time (RTT) and RTTVAR means RTT variation. Fig. 1 illustrates PARP, which

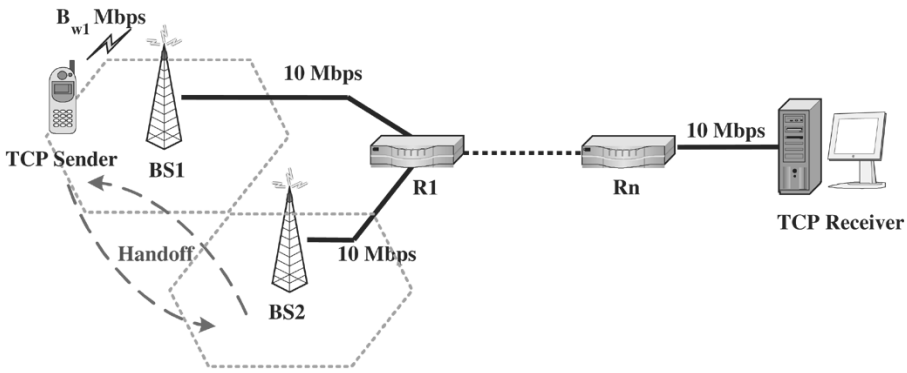

Fig. 2. Simulation environment.

is the period from T_{Pb} to T_{Pe}. Note that after a handoff, there is a short period called the ambiguous period (e.g., 1× SRTT). During this period, the reason for packet losses is ambiguous.

To determine P_H, the sender checks if the PARP of a lost packet is within the handoff period. If the PARP falls before $T_{\text{handoff\_finish}}$, P_H is set to Max (i.e., 100%). If the PARP is completely beyond the handoff period, P_H is set to Min (i.e., 0%).

If the PARP is in the ambiguous period, P_H is calculated as $P_H = ((T_{\text{handoff\_finish}} - T_{P_b})/(T_{P_e} - T_{P_b}))$.

P_C is calculated based on the *congestion history* α, which is the ratio of RTT to SRTT (i.e., RTT $= \alpha$ SRTT). Whenever segment losses are detected, *excluding* the losses occurring in the ambiguous period, the congestion history α is calculated as $\alpha_{\text{Sample}} = (\text{RTT\_ value / SRTT})$, and $\alpha_{i+1} = w_c \times \alpha_i + (1 - w_c) \times \alpha_{\text{Sample}}$, where w_c is the weighting factor, e.g., 0.2. The RTT_value is set to an Retransmission TimeOut (RTO) value if the reason of segment loss is retransmission timeout; otherwise, the RTT_value is set to the "last available" sampled RTT.

If RTT is less than SRTT, P_C is set to Min (i.e., 0%); if RTT is greater than α SRTT, P_C is set to Max (i.e., 100%). Otherwise, P_C is given by $P_C = (\text{RTT}/\alpha \times \text{SRTT})$.

B. Congestion Window Extrapolator (CWE)

The CWE is used to determine how the congestion window changes after a handoff. If no packet is lost due to congestion, and the sender has an empty usable window in the ambiguous period, the window size increases by the amount which should have occurred during the handoff. The detailed operation of CWE is described in Table I. Here an *acknowledgment interval* T_{ack} is calculated as $T_{\text{ack},i+1} = w_a \times T_{\text{ack},i} + (1 - w_a) \times T_{\text{ack}}$, where w_a is the weighting factor, $0 \le w_a \le 1$.

TABLE I
CONGESTION WINDOW EXTRAPOLATOR (CWE)

```
if ( (T_handoff_finish − T_handoff_start) > estimated_RTT )
        T_increase_interval = estimated_RTT ;
else
        T_increase_interval = T_handoff_finish − T_handoff_start ;

for ( i = 0; i < (int)( T_increase_interval / T_ack ); i++){
      if( cwnd_ < ssthresh_ ) /* slow start */
         cwnd_ += 1;
      else                    /* congestion avoidance */
         cwnd  += 1 / cwnd  ;  }
```

III. PERFORMANCE EVALUATION

In this section, we describe the simulations conducted to evaluate the performance of the proposed mechanism in a wireless network. The simulation environment is shown in Fig. 2. There are n routers between a BS and a wired receiver. The delay on each wired link is 50 ms. The proposed mechanism is implemented at the mobile sender, and the receiver runs TCP Reno. The mobile sender moves between BS1 and BS2 every 20 s. The data rates of the wireless links to BS1 and BS2 are set to B_{w1} and B_{w2}, respectively. The wireless delay D_w is variable. There is a file transfer protocol (FTP) connection between the TCP sender and the receiver. The FTP connection lasts for the duration of the simulation. The value of each parameter is listed as follows: $n = 3$, $B_{w1} = 10$ Mb/s, $B_{w2} = 10$ Mb/s, and $D_w = 100$ ms. The simulation results are generated using simulation tool ns-2. We compare the following mechanisms in the simulation: TCP Reno and Fast Retransmit with Freeze timer.

In Fig. 3, the proposed mechanism is compared with TCP Reno. Fig. 3(a) plots the congestion windows of the proposed mechanism and TCP Reno as a function of time. The curves of Reno and the proposed mechanism are plotted after a handoff (with simulation time 206 s). The curve marked "no-handoff" is only for comparison, showing the congestion window of a Reno connection when no handoff occurs. The proposed mechanism performs as if the handoff has never occurred, while Reno degrades when a handoff occurs. We then compare the throughputs of the proposed mechanism and TCP Reno. To better see the performance improvement of the proposed mechanism over TCP Reno, we define the performance "gain" as Gain = $(\text{Throughput}_{\text{proposed\_TCP}} - \text{Throughput}_{\text{Reno\_TCP}}/\text{Throughput}_{\text{Rene\_TCP}}) \times 100\%$, and plot the gain curve of each component over TCP Reno with different handoff periods in Fig. 3(b). The curve with diamonds corresponds to LC only (the middle one), the curve with rectangles is based on CWE only (the bottom one), and the curve with triangles accounts for both (the top one). We see that LC is more important than CWE in the proposed mechanism in terms of the throughput improvement for mobile TCP connections. The three curves of performance gains all exceed one, indicating that our mechanism outperforms TCP Reno.

Fig. 4 shows the hit ratio of LC, i.e., the percentage of "right guess" for packet losses by LC. Here T_{bad} indicates the duration

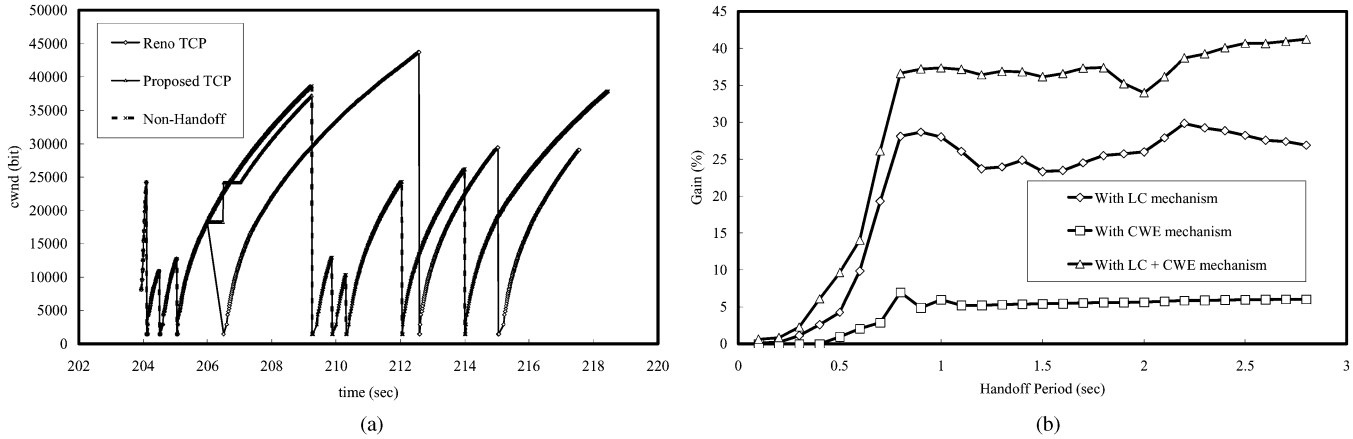

Fig. 3. TCP Reno versus proposed TCP. (a) Congestion window. (b) Throughput gain.

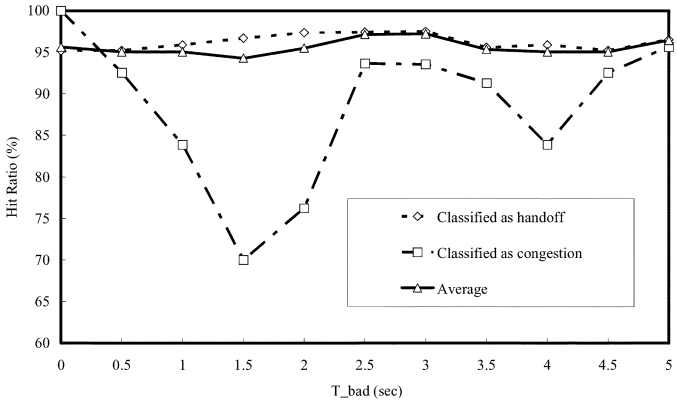

Fig. 4. Hit ratio of LC.

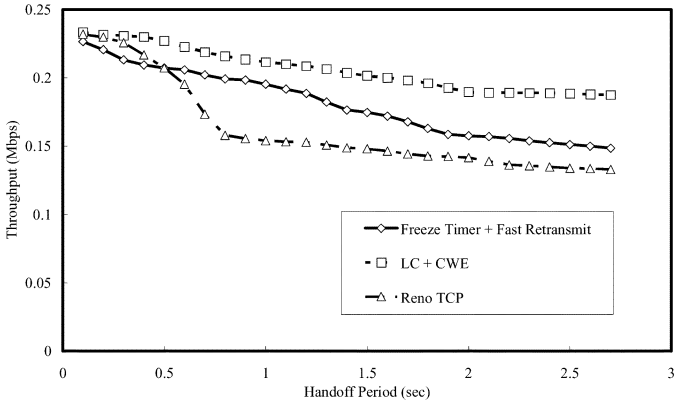

Fig. 5. Comparison of three mechanisms.

in which the wireless channel is in the bad state in every 10 s. When it is in the bad state, the wireless delay is set to twice the original delay of the link. The figure depicts that LC gives the right classification of packet losses over 95% of the time on the average. The wrong "guess" of packet losses being classified as congestion is mainly due to our treating transmission errors on the links in the same way as congestions in LC. Fig. 5 compares the performance of three mechanisms, including TCP Reno, the proposed mechanism, and a TCP variant having both freeze timer and fast-transmit mechanism. Since Freeze TCP and existing work focus on the receiver side, we implement the combined mechanism of Freeze TCP [8] with Fast Transmit [6] at the sender and compare it with our mechanism. The curve with squares corresponds to the proposed mechanism, the one with diamonds is the combined freeze timer and fast transmit, and the one with triangles is TCP Reno. Again, varying the handoff period from 0 to 3 s, we see that our mechanism outperforms both TCP Reno and Fast Retransmit with Freeze Timer.

IV. Concluding Remarks

In this letter, we have proposed an end-to-end mechanism which improves the performance of TCP for roaming users in mobile networks. The proposed mechanism is comprised of an LC and a CWE. LC determines why packets are not received after handoffs, and CWE determines how to adjust the congestion window after handoffs. We have conducted simulations to evaluate the performance of our mechanism. The results show that the proposed mechanism significantly improves TCP performance in mobile networks.

References

[1] A. Bakre and B. R. Badrinath, "I-TCP: Indirect TCP for mobile hosts," in *Proc. 15th Int. Conf. Distrib. Computing Syst.*, Jun. 1995, pp. 136–143.

[2] I. Rhee, N. Balaguru, and G. N. Rouskas, "MTCP: Scalable TCP-like congestion control for reliable multicast," in *Proc. IEEE INFOCOM*, 1999, pp. 1265–1273.

[3] H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *ACM Wireless Networks*, pp. 469–481, Dec. 1995.

[4] K. Brown and S. Singh, "M-TCP: TCP for mobile cellular networks," *ACM Computer Commun. Rev.*, vol. 27, pp. 19–43, 1997.

[5] P. Sinha *et al.*, "WTCP: A reliable transport protocol for wireless wide-area networks," in *Proc. ACM Mobicom*, 1999, pp. 301–316.

[6] R. Caceres and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," *IEEE J. Sel. Areas Commun.*, vol. 13, pp. 850–857, Jun. 1995.

[7] B. S. Bakshi, P. Krishna, N. H. Vaidya, and D. K. Pradhan, "Improving performance of TCP over wireless networks," in *Proc. Int. Conf. Distrib. Computing Syst.*, 1997, pp. 365–373.

[8] T. Goff, J. Moronski, D. S. Phatak, and V. Gupta, "Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments," in *Proc. IEEE INFOCOM*, 2000, pp. 1537–1545.