Regular Paper

TCPMobile: Improving TCP Performance in Wireless Networks

Vinh Dien Hoang,† Zhenhai Shao† and Masayuki Fujise†

Normal Reno TCP with slow start mechanism shows a very good performance for data communication in the wired network where packet losses occur mostly by congestions. However, its performance reduces significantly in the wireless network where losses occur also due to the lousy links and mobile host's handoffs. In this paper, we proposed a solution — TCPMobile to tackle this problem. Connection establishment, data transfer and handoff management are all presented in our solution. TCPMobile basically uses ideas in I-TCP to separate the connection between the wired and wireless network at the Foreign Agent (FA) of Mobile IP. Moreover, three important modifications in the TCP protocol are made at the wireless connection to mitigate the effect of handoff and the lousy wireless link. Simulations on NS2 simulator show a very good improvement in the network throughput. Another advantage of TCPMobile is that it only requires changes at the FA and slightly changes at the Mobile Host (MH) in the connection establishment phase.

1. Introduction

The telecommunication industry and the wireless technology develop very fast during the last decade and result in a lot of wireless networks such as GPRS, PHS, UMTS, IEEE 802.11x, ad hoc networks, etc. Certainly there's need for interconnecting these wireless networks with other existing networks. The de facto standard for internetworking between these networks is TCP/IP. However, there're currently two main problems for widely used Reno TCP. The first one is how to initiate and maintain connections when the mobile host in the wireless network moves frequently and changes its IP address. The second problem is how to provide a good performance TCP services in the wireless environment with lousy links and frequent handoffs.

To solve the first problem and provide connectivity between the Mobile Host (MH) and the Fixed Host (FH), Mobile IP had been proposed [1]. Using this scheme, a FH/MH can initiate the connection to other MHs using the Home Agent (HA) and the Foreign Agent (FA). Detail will be presented in Section 3.

As for the second problem, it's because during the development of the TCP protocol, timeout event occurs mainly due to the congestion at the intermediate routers. So the TCP protocol had been optimized for the reliable transmission network where losses are due to the congestion. It had resulted in Reno TCP [2] which is widely deployed nowadays. When a timeout occurs, Reno TCP assumes that this is due to the congestion at the intermediate nodes and activates its slow start algorithm. Although this algorithm works fine in the wired network, it shows a lot of problems when applying to wireless network where losses are also due to the lousy wireless links and the frequently movement of MHs.

In this paper a solution for this problem-TCPMobile is presented. Basically TCPMobile uses the idea of separating the TCP connection between the wired part and the wireless part at the FA. In addition, there important modifications on the TCP protocol for the connection in the wireless part are introduced to improve the TCP performance.

This paper is organized as follows. Related works are presented in Section 2. Mobile IP and the proposed solution (TCPMobile) are presented in Section 3. Section 4 is the performance evaluation of the proposed solution based on NS2 network simulator [3]. Basically, the simulation is used to evaluate the proposed solution over Reno TCP and M-TCP [4]. Finally, Section 5 is the conclusion of the paper.

2. Related Works

Many solutions had been proposed during the last 10 years to tackle the above mentioned problem. They can be classified into two big categories: keeps the end-to-end TCP connection, or separates the wired and wireless part of TCP connection (split TCP connection).

End-to-end TCP connection: Proposals

† Wireless Communications Laboratory, National Institute of Information and Communications Technology, Singapore

in this scheme either try to hide the losses due to the lousy wireless link and the frequent mobile handoff from the TCP sender (hence require changes on the receiver side) or use some algorithms at the sender side to differentiate between congestion losses and random losses (hence requires changes in the sender side).

Freeze-TCP [5] based on the fact that the MH somehow (e.g., from signal strength etc.) can predict the disconnection with the wired network beforehand so that it could advertise a zero window size ACK to the TCP sender to stop it from sending more packets or shrinking its congestion window. When the wireless connection resumes, the MH advertises the same ACK with a suitable window size to the sender and the transmission continues. This solution maintains the end-to-end TCP connection and requires changes only at the MH.

In TCP-MD&R [6], a MH bases on TCP-MD (moving detection algorithm) to predict handoff and TCP-R (registration) to freeze sender from sending more packets during the handoff. This solution improves the performance of TCP during handoff especially in frequent handoff.

However, when losses happened due to the lousy wireless link, the performance of Freeze-TCP and TCP-MD&R is degraded significantly since each time a packet is lost (easily happens in the wireless environment) the sender's congestion window is shrunk to one segment window size.

TCP k-SACK [7] is a little bit difference with TCP SACK in its congestion detection and avoidance algorithms. The TCP sender did not use the single packet loss as the indication of the congestion. Instead it uses a new parameter — *loss window*. If more than k packets transmitted in the *loss window* are lost then it's due to the congestion otherwise it's due to the lousy link. TCP k-SACK can achieve better performance compared with normal Reno TCP since it does not shrink its congestion window when less than k packets are lost. This solution requires changes at both sender and receiver.

Fast retransmission [8),9] uses triple ACKs to activate fast recovery algorithm of Reno TCP in the sender side. When detecting a packet loss, the MH sends triple ACKs to the sender. The sender upon receiving triple ACKs will resend the lost packet and keep its congestion window unchanged. This solution is simple and requires changes only in the MH side. However, each time fast recovery is activated using Fast retransmission, the TCP sender's threshold window is shrunk by half and this will reduce the connection performance later on.

TCP Veno [10] tries to distinguish the random losses caused by the lousy links with losses caused by the congestion and adjusts the slow start threshold to a suitable value (not 1 as in Reno TCP). TCP Veno requires change in the sender only.

However, TCP Veno, TCP k-SACK and Fast retransmission cannot effectively deal with the frequent handoff of the MH which is expected in the wireless network. And this will degrade the performance of the connection during and after handoff.

Split TCP connection: These proposals here break each TCP connection between the sender and the receiver into two separate connections: one TCP connection in the wired network and one TCP connection in the wireless network. The break point is usually in the Base Station (BS). I-TCP [11] is belonged to this category. In the wired network between the FH and the BS, a traditional TCP implementation (Reno TCP) is used. In the wireless network between the BS and the MH, a customized TCP protocol for wireless is used. BS will act as a bridge to forward packets between the two connections. This solution can solve all the problems since a new, customized TCP protocol is used in the wireless network. However it breaks the end-to-end TCP semantics.

M-TCP [4] is a very special solution in this group because it breaks the TCP connection at the BS but manages to keep the end-to-end TCP semantics. The original TCP connection from the FH to the MH will be split at the BS. When the BS receives a TCP segment from the FH, it sends the segment to the MH and waits for the MH's ACK. During that waiting time the BS does not acknowledge the segment with the FH. Only when the BS receives the ACK from the MH, it acknowledges the segment with the FH. By doing so, the TCP end-to-end semantic is kept. However, the most interesting feature of M-TCP is that the BS always keeps the last byte unacknowledged. It means if the MH had acknowledged with the BS k bytes, the BS only acknowledged with the FH $k-1$ bytes. When the BS senses that the MH was disconnected (e.g., during handoff), it will send an ACK packet with zero-window size to the FH for the last byte — the kth byte. This ACK will freeze the TCP sender at the FH and prevent it

from shrinking its transmission windows. When the BS-MH connection is restored, the BS will send the same ACK with suitable window size to the FH to reactivate its TCP sender and continue with the data transmission.

It's observed from the M-TCP operations that the BS does not acknowledge the TCP segment it received immediately. It has to wait for the ACK from the MH first. This will add delay to the connection in the FH-BS part. Moreover, when the MH finishes handoff and restores the BS-MH connection. Data transmission also does not resume immediately because it has to wait for the BS to "wake-up" the TCP sender at the FH. M-TCP also did not make any significant changes on the TCP protocol at the wireless connection except on handling handoffs. These factors will reduce the overall M-TCP connection's performance.

Snoop [12] can be considered as belonging to the first category. However, it needs information from transport protocol (TCP) to operate and sometimes it was considered as belonging to the link-layer protocol category. In fact, snoop can be classified as somewhere between the two above categories. By using an agent residing in the intermediate node (usually the BS) to read the TCP header, packets are cached and retransmitted to the MH if they are lost. However, like Fast retransmission, TCP k-SACK, Snoop is unable to effectively deal with frequent handoffs. Its performance will suffer after a longtime disconnection.

The above analyses are the reasons for us to find a TCP solution that can maintain a good performance in the lousy wireless link environment with frequent handoffs. It's main objective of this paper.

3. TCPMobile

To provide complete TCP services for the MH, we have to assure that the MH can establish and maintain the connection with other hosts and vice versa when the MH connects to the network. But due to the MH frequent movement, it can be expected that IP address of the MH will be changed as MH moves to a new place. With this new IP address, the MH still can send data to the other hosts but the other hosts can not do so because there's no mechanism to inform them about the MH's new IP address. Mobile IP [1] is the best solution up till now to tackle this problem.

Using Mobile IP, the MH is assigned to a home network that is represented by a Home Agent (HA). Each MH has a static IP address belonged to the home network. Other hosts will communicate with the MH using this static IP address. When the MH moves to other networks, it will obtain another IP address called IP care-of address (CoA) from the FA in that network. The FA in that network in turn will inform the MH's HA about the presence of its MH in the foreign network. If the foreign network does not have a FA or all FAs are busy, the MH will act as the FA and obtain its so call co-located Care of Address (using DHCP, etc).

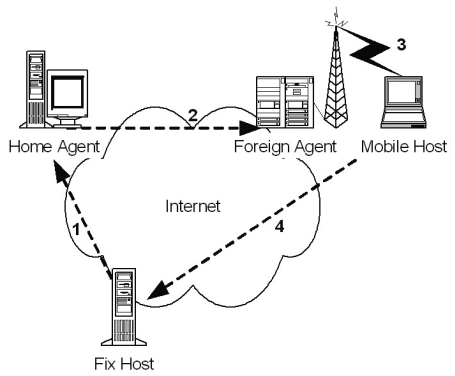

Fig. 1 Mobile IP operations.

Figure 1 illustrates how mobile IP work. When a FH wants to send data to a MH, it sends the IP packet to MH home IP address (1). This packet will be routed to the MH's home network and intercepted by the HA. The HA will encapsulate this entire datagram and forward it to the MH's CoA address (or co-located Care of Address). This encapsulated packet will eventually reach the FA (2) where the MH resides. The FA will then decapsulate it and send the original IP datagram to the destination MH (3).

If the MH wants to send a packet to other hosts (could be FH or MH), it will send this packet to FH's IP address or MH's home IP address. This packet will then be routed by foreign network routers to the FH or MH's home network (4).

Since the MH is moving frequently, there should be a mechanism for the MH to detect its movement. The MH can do so by listening to the agent advertisement messages periodically broadcasted by the FA. Based on the information on these messages, the MH can determine whether it has moved to a new foreign network or not.

Using mobile IP, a TCP connection can be established between a host and a MH without being afraid of IP address changes of the MH. However, this TCP connection's performance will be affected as a normal TCP does because of the packet losses in the wireless network and during MH handoff.

It can be seen from the Mobile IP operations that the FA has to strip off the HA's encapsulation to recover the original datagram from the FH and forward it to the MH through the foreign network which may has different MTU (Maximum Transmission Unit), etc. Moreover, all packets arrived at the MH are forwarded by the FA. This is true even with the use of binding update to eliminate the triangle routing problem in Mobile IP. In addition, FA is belonged to the foreign network (in most case FA is BS) which is usually only one or two hops away from the MH. That's reason we proposed to break the connection between the FH and the MH at the FA.

In the proposed solution (TCPMobile), separated connection idea used in I-TCP is adapted and integrated with Mobile IP at the FA. Whenever the FA receives a datagram from the FH through the HA, it will send an ACK for this datagram on MH's behalf. This datagram will be stored in the FA's buffer and eventually sent to the MH through a separated connection. So only the MH and the FA are aware of this separation. It's true that the FA is acting as a proxy for the MH. In the wireless connection between the FA and the MH, a modified TCP implementation is used to improve the performance of the connection in the wireless network. Moreover, an effective handoff mechanism is introduced. Using this handoff mechanism, the FA will actively freeze the FH's TCP sender whenever it detects a handoff in progress. Following are the detail operations of TCPMobile.

3.1 Connection Establishment and Data Transfer

If the FH initiates the connection, it will send a TCP segment with the SYN bit on and the ACK bit off to the MH through the HA. When this segment reaches FA, FA will issue an ACK to the FH on behalf of the MH and the connection between FA and FH is established. The FA then establishes a separate connection to the MH. The connections are now ready for data transfer.

If the MH initiates the connection with another host (FH or another MH), it will send a TCP segment with the SYN bit on and the ACK bit off to the FH through the foreign network router (may not be the FA). This connection request segment will eventually arrive at the FH and an ACK is issued to the MH through the HA and FA. Upon receiving this ACK, the FA will reply to the FH on MH's behalf and the connection between FA and FH is established. The FA then establishes a separate connection with the MH. After this connection has been established, the FA will act as a bridge and forward data between the two connections.

To mitigate the lousy wireless links, two simple but important modifications are applied at the Reno TCP for the connection between the FA-MH.

- When the FA receives two duplicate ACKs from the MH, it will assume that the packet has been lost and retransmit it. It will not enter fast recovery (Reno TCP enters fast recovery when it receives three duplicate ACKs). When the number of duplicate ACKs seen at the FA reaches 5, TCP sender at the FA enters fast recovery. This modification ensures that the lost packet is retransmitted in a timely manner (after two duplicate ACKs). This modification also ensures that if the congestion happened, the TCP sender in the FA would reduce its congestion window and enter congestion avoidance phase (after 5 duplicate ACKs).
- When timeout happened, the TCP sender at the FA retransmits the timeout packet. It does not enter the slow start phase as Reno TCP does. During that time, if there's another timeout happens and no new ACK came, the TCP sender will enter the slow start phase to avoid congestions at the intermediate nodes.

3.2 Handoff Management

The FA detects handoff in progress after a timeout period from the MH (currently we choose 4 x *FA-MH round trip delay*) or after receiving a handoff request message from another FA. When detecting that handoff is in progress (**Fig. 2**), the FA will restore the TCP sockets associated with the MH to its previous state, send a zero window size ACK to the FH to freeze FH's socket and wait for the handoff request message from the new FA. When the handoff request message comes (2), the old FA will copy all of the socket's status associated with the MH involved to the new FA (3). The new FA then issues an ACK with non-zero window size to the FH and reactivates the con-

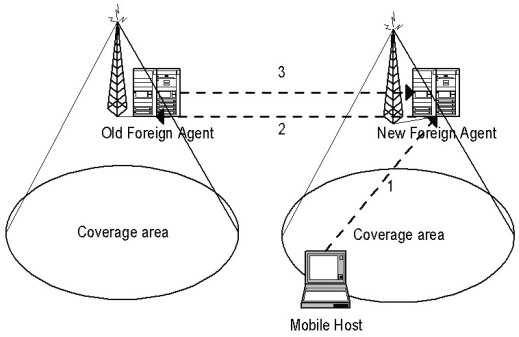

Fig. 2 Handoff.

nection. The FH isn't aware of this handoff process.

When the MH moves to a new cell, it could detect its movement based on agent advertisement messages broadcasted by the new FA or it can issue its own solicitation message to explicitly ask for advertisement messages after a timeout period. Upon receiving these messages, the MH could detect its movement and liaise with the new FA to complete its Mobile IP registration process. The MH then informs the new FA about its previous connections and the old FA's address (1). The new FA based on that information will liaise with the old FA to create exactly the same TCP sockets associated with the MH in the old FA (2, 3). After handoff finishes, the old FA will delete all TCP sockets associated with the old MH.

Unlike [6], in our TCPMobile solution, the FA will send a zero window size ACK to freeze FH's TCP sender after a timeout period. If the MH sends a zero window size ACK to the FH after detecting its movements as in[6], it's too late since it usually takes more than a round trip time to detect the MH movement and the FH already shrank its congestion windows.

4. Performance Evaluation

Simulations on the NS2 simulator [3] are used to evaluate the effectiveness of the proposed TCPMobile solution. To achieve a fair evaluation, Reno TCP and M-TCP will be studied together with TCPMobile in two different scenarios respectively.

The following metrics are used for the performance comparison:

Throughput: Number of packet received at the destination over the simulation duration.

End to end packet delay: The average delivery delay of the packet from the source to the destination. This delay is calculated from the time the packet is created by the application at the source to the time the packet is received by the application at the destination. It includes buffering time, transmission time, processing time at the source, intermediate nodes and the destination.

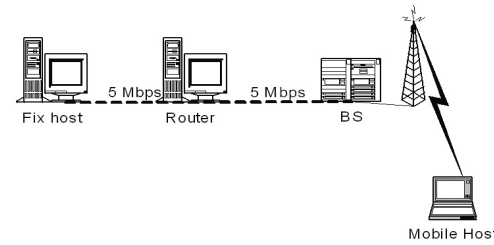

Fig. 3 Reno TCP simulation model.

In the first scenario, TCPMobile and Reno TCP are simulated in the model shown on **Fig. 3**. This model consists of a FH, a router, a BS and a MH. In this model, the FH and the BS also act as the HA and FA respectively. TCP segments are sent from the HA (FH) to the FA (also BS). Upon receiving them, the FA will buffer and send ACKs back on behalf of the MH and be responsible for delivering these TCP segments to the MH through another separated connection.

The connection from the HA to the FA is in the wired network with 5 Mbps bandwidth. The connection from the FA to the MH is in the wireless network with 1 Mbps bandwidth. The time delay between the HA and the Router and between the router and the FA are 2 ms.

Packets transmitted through the wireless link will be lost with the probability of p. In the wireless environment, packet loss can happen at any time. So by using loss probability p, the lousy wireless link could be modeled. By varying p, effects of the lousy wireless link on the TCP's performance can be studied. In addition, the wireless link will be disconnected for two times, each time 2 seconds during the simulation. This is used to study the protocol ability to recover from the long time disconnection. All simulations lasts 550 seconds.

TCP parameters:
Initial windows size: 20
Initial Threshold size: 20
Packet size: 1040 bytes
ACK packet size: 40 bytes
TCP Buffer: 50 packets

Figure 4 shows the throughput of Reno TCP

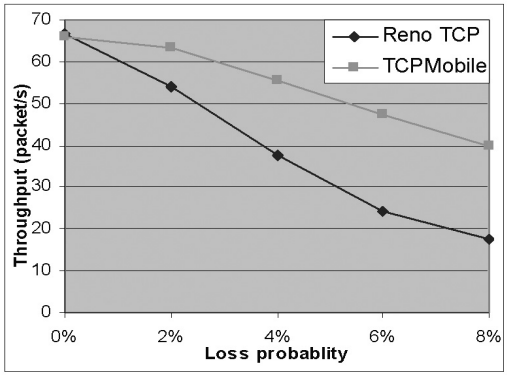

Fig. 4 Reno TCP-TCPMobile throughput vs. loss probability.

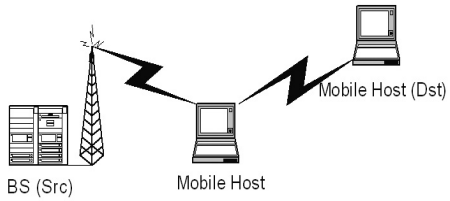

Fig. 5 M-TCP simulation model.

and TCPMobile when loss probability in the wireless network increased. This simulation proved that Reno TCP throughput will be suffered especially in the high loss environment. Each time the packet is lost, Reno TCP reduces its threshold and congestion window by half and slows its data transmission, in this case, unnecessarily. TCPMobile, on the other hand, retransmits the lost packet and slowly adjusts its threshold and congestion window to mitigate the lousy link, and as the results, impressively improves the throughput, especially in the high packet loss scenario.

The second simulation is used to compare the performance of the proposed solution with a prominent protocol and M-TCP has been chosen because it has many advantages and uses the same philosophy as TCPMobile. In this scenario, the difference and the behavior of M-TCP and TCPMobile will be studied in the model shown on **Fig. 5**. This model only consists of the wireless part because the most significant differences between TCPMobile and M-TCP are on this part. Doing so also simplifies the simulation. Using this model, a wireless network consists of a BS, two MHs will be simulated. TCP segments will be transmitted from the source (the BS) to the destination (the MH two hops away) using M-TCP and TCPMobile.

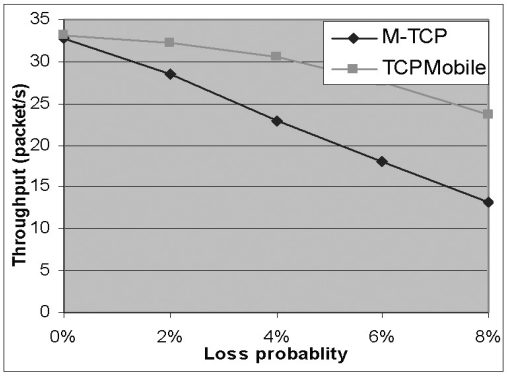

Fig. 6 M-TCP-TCPMobile throughput vs. loss probability.

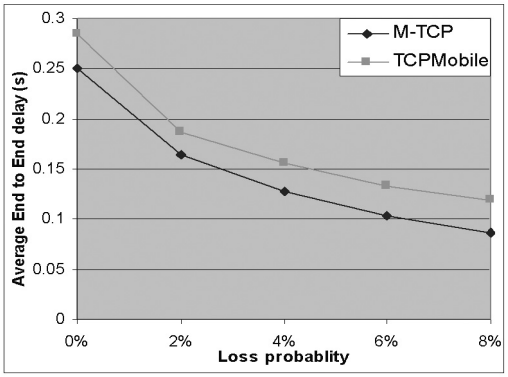

Fig. 7 M-TCP-TCPMobile average end-to-end delay.

All other parameters are the same as in the previous scenario.

The throughput of M-TCP and TCPMobile are shown in the **Fig. 6**. It could be seen that both M-TCP and TCPMobile are affected by the packet losses. However, when the losses increase, M-TCP throughput reduces much faster than that of TCPMobile. When loss probability reaches 8%, there's quite big different between throughput of TCPMobile and M-TCP as shown in Fig. 6. This is because TCPMobile, with two modifications on handling timeouts and duplicate ACKs, retransmits the loss packet more timely and adjusts the congestion window more precisely with the current network situation. These two modifications effectively improve TCPMobile throughput even in a high loss network. The throughput of TCPMobile is almost double that of M-TCP when losses increase to 8%.

However, the higher throughput of TCPMobile comes with a price. That's the higher end-to-end delay. **Figure 7** shows the compari-

son of the end-to-end delay between the TCP-Mobile and M-TCP. TCPMobile suffers higher end-to-end delay even in the low loss network. There're two contributing factors to this higher delay. The first factor is that the throughput of TCPMobile is higher. This higher throughput adds more delay to the transmission time and processing time at the source, destination and intermediate nodes. The second factor is the way TCPMobile sending its TCP segments. After spending some time adjusting the congestion window and threshold window, TCP-Mobile will usually send the TCP segments in burst. This burst adds more stress to the intermediate nodes, wireless bandwidth resource and causes higher end-to-end delay.

The above simulation results clearly show that TCPMobile's performance is very good compared with Reno TCP and M-TCP especially in the high loss wireless network.

5. Conclusion

Mobile IP and different TCP schemes used for the wireless network have been studied in this paper. Because of the nature difference of the packet loss in the wired and the wireless network, the best solution for improving TCP performance in the wireless network is that it should have different TCP implementations. And the FA is the most suitable point to split the TCP connection so that a different TCP protocol for the connection from the FA to MH could be implemented. Details of the proposed solution - TCPMobile are then introduced which include all connection establishment, data transfer and handoff management phases. This solution also uses a modified TCP implementation for the wireless part which differs with Reno TCP in the way to handle duplicate ACK packets and timeout events. Simulations on the NS2 simulator are used to evaluate the proposed solution over Reno TCP and M-TCP. The simulation proves that the proposed solution achieves better performance than Reno TCP and M-TCP do, in term of throughput. However, more research should be carried out in the connection between the FA and MH to reduce its higher packet end-to-end delay. And it will be the objective of our future research.

References

1) Perkins, C.: IP mobility support, RFC 2002 (1996).
2) Stevens, W.R.: TCP slow start, congestion avoidance, fast retransmission, and fast recovery algorithms, *IETF*, RFC 2001 (Jan. 1997).
3) Fall, K. and Varadhan, K.: NS Manual, The VINT Project, http://www.isi.edu/nsnam/ns/doc/
4) Brown, K. and Singh, S.: M-TCP: TCP for Mobile Cellular Network, *ACM Computer Communication Review*, Vol.27, No.5 (1997).
5) Goff, T., Moronski, J. and Phatak, D.S.: Freeze-TCP: A true end to end TCP enhancement mechanism for mobile environments, *IEEE Proc. 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol.3, pp.1537–1545 (2000).
6) Kwon, J.W., Park, H.D. and Cho, Y.Z.: An Efficient Mechanism for Mobile IP Handoffs, *IEICE Trans. Commun.*, Vol.E85-B, No.4, pp.796–801 (2002).
7) Chrungoo, A., Gupta, V., Saran, H. and Shorey, R.: TCP k-SACK: A simple protocol to improve performance over lossy links, *IEEE Global Telecommunications Conference*, Vol.3, pp.1713–1717 (2001).
8) Caceres, R. and Iftode, L.: Improving the performance of reliable transport protocols in mobile computing environments, *IEEE J. Select. Areas Commun.*, Vol.13, pp.850–857 (1995).
9) Nakanishi, J., Tsukamoto, K. and Komaki, S.: Proposal of continuous-FRT method for TCP/IP access in spot type DERC system, *Proc. 3rd International Workshop on ITS Telecommunications*, pp.269–273 (2002).
10) Fu, C.P. and Liew, S.C.: TCP veno: TCP enhancement for transmission over wireless access networks, *IEEE J. Select. Areas Commun.*, Vol.21, No.2, pp.216–228 (2003).
11) Bakre, A. and Badrinath, B.R.: I-TCP: Indirect TCP for mobile hosts, *IEEE Proc. 15th Int. Conf. Distributed Computing Syst.* (*ICDCS*), pp.136–143 (1995).
12) Balakrishnan, H., Seshan, S. and Katz, R.H.: Improving reliable transport and handoff performance in cellular wireless networks, *ACM Wireless Networks*, Vol.1, pp.469–481 (1995).

(Received April 10, 2004)
(Accepted July 1, 2004)

Vinh Dien Hoang was born in HaiHung, Vietnam on October 11, 1974. He graduated from Hanoi National University, Vietnam, with a BSc degree in mathematics and informatic in 1996, and a MSc degree in mathematics 1999. In 2002, he received the MSc in "Communication software and Networks" from School of EEE, Nanyang Technological University, Singapore. He is currently working as research engineer for the Wireless Communications Laboratory, National Institute of Information and Communications Technology (NiCT), Singapore office. His research interests include TCP, Routing, MANET and Software Defined Radio.

Zhenhai Shao was born in Jiangsu Province, China, in December, 1971. He received the B.E. degree from the Nanjing Normal University, Nanjing, China, in 1994, and received the M.S. and Ph.D. degrees from the Southeast University, Najing, China, in 1997 and 2000, respectively. From 2000 to 2001, he was with National University of Singapore as research fellow. From 2001 to April 2003, he was with Nanyang Technological University as research fellow. Currently, He is a manager and research scientist in Wireless Communications Laboratory, NICT Singapore representative office, National Institute of Information and Communications Technology of Japan. His research interests include microwave /millimeter-wave subsystem, passive/active devices and circuits, numerical procedures of FDTD, TLM, TDFEM and DSC method for passive microwave components, software design. He also is in charge of researches about software defined radio, radio over fiber and ad hoc network. He authored over 30 paper published in international journals and conference proceedings. Dr. Shao is a member of IEEE. He has been a reviewer for the IEEE Transactions on Microwave Theory and Techniques and for the IEEE Microwave and Wireless Components Letters.

Masayuki Fujise was born in Fukuoka, Japan, on December 8, 1950. He received the B.S., M.S. and Dr. Eng. degrees, in communication engineering from Kyushu University, Fukuoka, Japan, in 1973, 1975 and 1987, respectively and the M. Eng. degree in electrical engineering from Cornell University, Ithaca, NY, in 1980. He joined KDD in 1975 and was with the R&D Laboratories being engaged in research on optical fiber transmission measurements. In 1990, he joined ATR Optical and Radio Communications Research Laboratories Kyoto as a department head. Then, he joined CRL in 1997 and he is now the director of Wireless Communications Laboratory, NICT in Singapore. Dr. Fujise is the recipient of the Jack Spergel Memorial Award of the 33rd International Wire & Cable Symposium in 1984 and he is a member of the IEICE Japan and the IEEE.