PAPER

Performance Improvement by Packet Buffering in Mobile IP Based Networks

Doo Seop EOM[†], Masashi SUGANO[††], Masayuki MURATA[†††], *and* Hideo MIYAHARA[††††], *Regular Members*

SUMMARY It is well-known that TCP often experiences severe performance degradation in mobile networks since packet losses not related to network congestion occur frequently due to host mobility. In this paper, we propose a new packet buffering method to address such a problem without the scalability problem in Mobile IP based networks. For this purpose, we first investigate the performance of TCP Tahoe without considering packet buffering through the simulation. Our simulation result shows that in most cases, the smooth handoff by the route optimization extension of Mobile IP standard cannot prevent the degradation of TCP performance due to handoffs, although it is designed to reduce the number of packets dropped during the handoff. It also shows that in utilizing the route optimization extension, the TCP performance sometimes becomes worse even than the case of the base Mobile IP unless its smooth handoff makes less than four packets be dropped during the handoff. Such results mean that at least for TCP, the smooth handoff is not useful unless the route optimization extension supports the buffering method, which makes handoffs be transparent to transport layer protocols by recovering the packets dropped during the handoff. We then investigate the effects of packet buffering on the performance of TCP. We modify the route optimization extension in order to support packet buffering at the base station, but it is a very minor change. Finally, we discuss some problems that should be addressed to recover the packets dropped during the handoff by the buffering method without giving a worse impact on the performance of TCP, and propose our solution to solve those problems.

key words: *mobile network, Mobile IP, TCP, packet buffering*

1. Introduction

With recent remarkable developments of wireless technologies and very fast proliferation of mobile hosts in past a few years, it becomes a realistic scenario that mobile host users can freely move from place to place while continuing communications. To support such a scenario in the Internet, the IETF (Internet Engineering Task Force) has designed a Mobile IP protocol. Mobile IP, acting as an inter-subnetwork handoff protocol within an IP layer, offers a mechanism for seamless handoff, but packet loss is unavoidable in the mobile network environment when a mobile host user moves into another subnetwork from a current one [1]. The problem is that TCP, the upper layer protocol than IP, was not originally designed for such an environment. That is, it interprets the packet losses as a sign of network congestion. It is even true when packet losses happen due to handoff, which results in unnecessary congestion control. Then, TCP suffers from severe performance degradation especially when a mobile host user visits many subnetworks during the connection.

Such a problem was first pointed out in [2], and several research works have been followed. See, e.g., [3]–[5]. However, no easy solution is still available until now. Some solutions attempt to solve the problem by modifying TCP. One such a solution can be found in [6] where the TCP receiver on the mobile host explicitly notifies the handoff event of the TCP sender so that an appropriate action can be taken by the TCP sender. It avoids degradation of TCP performance since the TCP sender can know that the packet loss takes place by the handoff event, not by the congestion occurrence. However, it apparently violates the layering concept of network protocol stack since the TCP receiver can never recognize the handoff event without help of underlying link or network layer protocols. Furthermore, the solution requires modification of TCP at both sides; not only at the TCP receiver of the mobile host, but also at the TCP sender connected to the wired-networks. It is very unlikely that such a modified version of TCP is installed on all correspondent hosts in the near future.

Another solution is the fast retransmit method [2], which forces the mobile host to send three duplicate ACKs needed to initiate Fast Retransmit at the correspondent host as soon as it completes the handoff. By this mechanism, the correspondent host can successfully retransmit lost packets without waiting for long retransmission timeout due to coarse timer resolution. The major problem is that this approach is useful only when one or two packets are dropped during the handoff and the number of packets arriving at the mobile host after handoff is too small to generate three duplicate ACKs [3]. Even in that case, the TCP sender performs unnecessary congestion control. Furthermore, it also requires a modification of TCP at the mobile host while modification is minor compared to the previous

Manuscript received December 21, 1998.
Manuscript revised June 14, 1999.

[†]The author is with the School of Electrical Engineering, Korea University, Seoul, 136-075, Korea.
[††]The author is with Osaka Prefecture College of Health Sciences, Habikino-shi, 583-8555 Japan.
[†††]The author is with Advanced Networked Environment Research Division, Cybermedia Center, Osaka University, Toyonaka-shi, 560-0043 Japan.
[††††]The author is with the Graduate School of Engineering Science, Osaka University, Toyonaka-shi, 560-8531 Japan.

approach. It also requires an indication from the lower link or network layer handoff protocol such that TCP knows the occurrence of the handoff.

We thus take another approach to pursue a seamless handoff in this paper. If we can achieve the seamless handoff, it gives an advantage that application programs having been designed for wired networks can also be used on the mobile hosts without any modification. A most promising way to realize the seamless handoff is to store the packets at the base station as a provision for the dropping of packets during the handoff event. Here, we classify packet buffering methods into two categories; an unicast-based buffering method and a multicast-based buffering method. In the former method, only the current base station for the mobile host performs packet buffering. It forwards the buffered packets to the new base station to which the mobile host is connected after handoff, immediately after the address of the new base station is informed to the previous base station.

On the other hand, in the latter method, all adjacent base stations of the current base station perform packet buffering with anticipation of future handoff of the mobile host. For this purpose, the packets destined for the mobile host are forwarded to all adjacent base stations in addition to the current base station by using multicasting routing. Therefore, there is no need to forward the buffered packets to the new base station at the time when the handoff actually takes place. It thus makes the handoff latency be shorter than that of the unicast-based buffering. However, for doing so, the overhead of buffering is increased and more complicated multicasting routing is necessary, in the case of the multicast-based buffering.

The buffering method has already been addressed in past literature. One such an example can be found in the split-connection protocol [7], [8] while its main purpose is to avoid the TCP performance degradation due to the high error rate of the wireless link. In the split-connection protocol, an end-to-end TCP connection is broken into the wired part (the fixed host to the base station) and the wireless part (the base station to the mobile host), and all information of the TCP connections opened at the current base station has to be handed over to the new base station when the handoff takes place. It includes the buffered packets. That is, the unicast-based buffering is used in that protocol [3], [9], [10]. In [1], the authors investigated impacts of the buffer size and beacon periods on the number of packets dropped during the handoff under the unicast-based buffering method.

The snoop protocol is another example to use the buffering method [5], in which every TCP packet of both directions is monitored by the snoop agent in the base station for the purpose of local retransmissions to the mobile host. The authors incorporated the multicast-based buffering method to improve the TCP performance against the handoffs as well as the high error rate of wireless link. However, all of those works only consider an intra-subnetwork handoff, which occurs within a subnetwork, and did not investigate impacts of packet buffering on the performance of TCP sufficiently. As we will show later, some problems must be addressed to support buffering successfully.

In [11], the authors compared the above solutions with link-layer solutions using forward error correction (FEC) or automatic repeat request (ARQ) techniques, for TCP over erroneous wireless link (without considering the handoff). They concluded that the most efficient one is a well-tuned link-layer solution which uses ACK of TCP for avoiding unnecessary TCP level retransmissions and selective ACK for efficient retransmission over the wireless link. It means that complicated solutions such as the split-connection protocol are not necessary. However, it also implies that the TCP performance degradation due to handoffs should also be handled independently of higher layer protocols so that it naturally fits into the layered structure of network protocols. Therefore, we believe that by combining a well-tuned link-layer solution and the buffering method that we address in this paper, two causes of the TCP performance degradation in mobile environments can be resolved most efficiently. Then, the existing applications can be used on the mobile host without any modification. Assuming that a well-tuned link-layer solution is supported to handle the high error rate of the wireless link, we focus on the effective buffering method to handle handoffs in the current paper.

In the IETF, buffering at one or more foreign agents is currently under consideration for more robust packet delivery to the mobile host [12]. Our method presented in this paper can offer the solution.

2. Packet Buffering in Mobile IP Based Network

In this section, we briefly summarize the current Mobile IP standard in Sect. 2.1, and then present our proposal to support packet buffering in the Mobile IP based network in Sect. 2.2. A simple analysis is also presented to estimate the buffer requirement necessary to recover all the packets dropped during the handoff. We consider the network configuration of Mobile IP based network shown in Fig. 1 where we assume that the router in each subnetwork also plays the role of FA (Foreign Agent).

2.1 Overview of Mobile IP

We first provide an overview of the base Mobile IP protocol [13] which can support host mobility without modification to existing routers or correspondent hosts. Then, we explain the Mobile IP with route optimization extension which is designed to solve the well-known triangle routing problem [12], [14] in the base Mobile IP

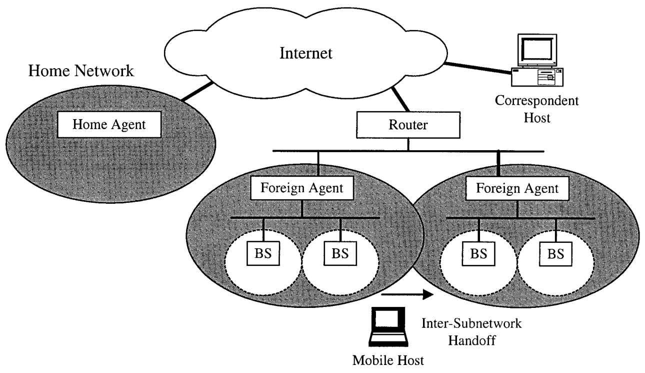


Fig. 1 Network configuration of Mobile IP based networks.

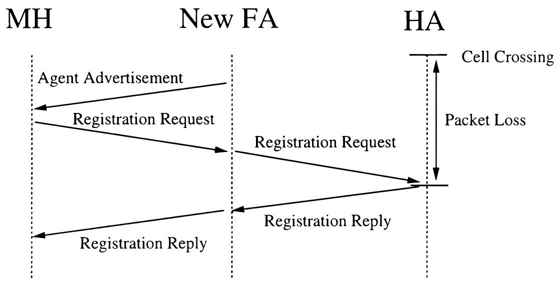


Fig. 2 Base IETF Mobile IP protocol.

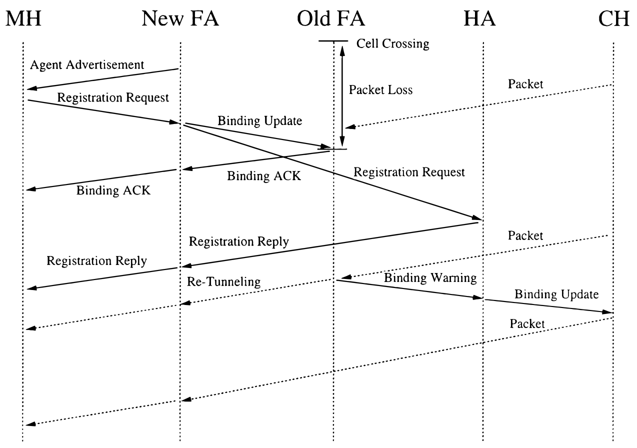


Fig. 3 Mobile IP with route optimization extension.

protocol.

In this paper, we proceed our discussion without explaining the functional entities and terms adopted in the mobile IP specification. Refer to [13], [14] for introductory terminologies. In the base Mobile IP protocol, the HA (Home Agent) in the home network of the mobile host intercepts the packets destined for the mobile host using proxy ARP, and then delivers them to the mobile host's current attachment point to the Internet using tunneling. The current attachment point is defined by an IP address called the care-of-address. There are two different types of care-of-address; a foreign agent care-of-address is an address of a FA with which the mobile host is registered, and a co-located care-of-address is an externally obtained local address which the mobile host has associated with one of its own network interfaces. We consider the foreign agent care-of-address in this paper.

As shown in Fig. 2, when a mobile host (MH) moves into a new foreign network, it receives the agent advertisement message including the care-of-address (i.e., the address of the new FA) from the new FA. At this time, the mobile host can realize that it moves into the new foreign network, and then sends the registration request message requiring registration of the new care-of-address to the new FA. The new FA relays it to the HA of the mobile host so that the HA delivers the packets to the new FA instead of the old (previous) FA.

In the Mobile IP with route optimization extension [15], when the new FA receives the registration request message from the mobile host, it sends the binding update message to the old FA in order to inform the new care-of-address, in addition to relaying the registration request message. See Fig. 3. Each time the old FA receives a packet from the correspondent host (CH), it has a responsibility to forward the packet to the new FA. Also, it sends the binding warning message to the HA because the correspondent host cannot know the new care-of-address until the HA informs it. When the HA receives the binding warning message, it sends the binding update message to the correspondent host in order to inform the new care-of-address. After receiving the binding update message, the correspondent host can send packets to the new FA instead of the old FA. Thus, the above process, called a smooth handoff, can reduce the number of packets dropped during the handoff because in general the new FA is likely to be farther from the HA than from the old FA.

2.2 Proposed Protocol to Support Packet Buffering

The number of packets dropped during the handoff can be reduced by the smooth handoff. However, there is still a possibility that in-flight packets are dropped until the old FA receives the binding update message from the new FA. This packet dropping gives severe impacts on the performance of TCP in mobile networks, as having been explained in Sect. 1. In the IETF, packet buffering at one or more FAs is discussed to resolve this problem [12].

We first consider the case where only one FA performs buffering. That is, only the current FA performs packet buffering with anticipation of future handoff of the mobile host. The current FA sends the buffered packets to the new FA when receiving the binding update message from the new FA. This method does not require any modification of the Mobile IP with route optimization extension. It is because buffering is performed only at the current FA. However, there is a scalability problem in this method because FA must cover all the TCP connections of the mobile hosts communicating with the IP hosts at the outside of the subnetwork. As packet buffering is performed in parallel at more FAs, the scalability problem becomes more serious. It is also difficult to support the handoff between two cells in the same subnetwork (i.e., intra-subnetwork/local handoff) when buffering is performed at FA.

At the data link-layer, ARQ protocol uses the retransmission buffer to perform error control on the wireless link. It can be used for recovering the packets dropped during the handoff if buffering is performed at BS (Base Station) [1]. In that case, only the packets not acknowledged by the mobile host are forwarded to the new BS after handoff. That is, the mobile host never receives duplicate packets due to buffering. As will be shown in the next section, such duplicate packets trigger an unnecessary Fast Retransmit. Thus, the performance of TCP cannot be improved by buffering solely. However, if buffering is performed at FA, such duplicate packets are unavoidable. It is possible that FA manages the buffer for recovering packet losses (due to handoff) by using the ACK of TCP, but monitoring of TCP ACKs introduces overheads at FA.

From the above reasons, we propose an unicast-based buffering method where buffering is performed only at the current BS. In this method, the packets buffered in the old BS are first sent to the old FA. After that, the old FA forwards them to the mobile host through the new FA. We note that in the route optimization extension, if the packets destined for the mobile host are wrongly tunneled to the old FA, it forwards those packets to the new FA by re-encapsulating them with the right care-of-address (i.e., the address of the new FA). Therefore, the problem in the unicast-based buffering method becomes how the old BS sends the buffered packets to the old FA. For this purpose, we introduce the *forward* message (see Fig. 4), which serves as a link-layer control message between FA and BS. It contains the link-layer address of the mobile host to identify the buffered packets for the mobile host. When the old FA receives the binding update message from the new FA, it sends the *forward* message to the old BS to request forwarding of the buffered packets. When those packets arrive at the old FA, it can handle them as the packets wrongly tunneled to it. That is, it forwards the buffered packets to the new FA by re-encapsulating them with the address of the new FA. Therefore, with minor modification of the route optimization extension, the proposed buffering method can solve the problems that may happen when buffering is performed at FA.

On the other hand, the Mobile IP standard [13] recommends to limit the maximum sending rate of the agent advertisement message to once per second for reducing the network load caused by the agent advertisement message. Therefore, even if the agent advertisement message is broadcasted (or multicasted) at the maximum sending rate by the new FA, the mobile host cannot receive it during one second after moving into the new foreign network in the worst case. It means that during that time, in-flight packets destined for the mobile host are dropped because the mobile host cannot send the registration request message until receiving the agent advertisement message. For this problem, we make the Mobile IP with route optimization extension be incorporated with the local handoff protocol as shown in Fig. 4. After receiving the beacon message which plays the same role as the agent advertisement message, the mobile host sends the handoff request message to the new BS. The new BS then sends the notification message to the new FA for requesting the agent advertisement message. Upon receiving the notification message, the new FA sends the agent advertisement message to the new BS.

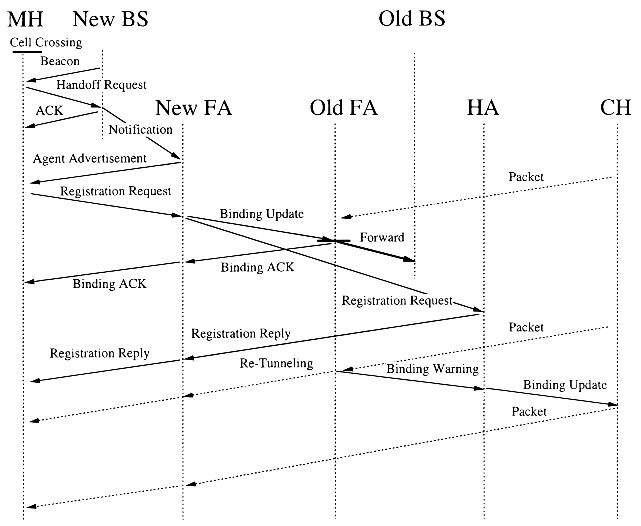


Fig. 4 Modification of the Mobile IP with route optimization extension to support buffering at a base station.

By this method, the mobile host can receive the agent advertisement message more quickly, compared to the method which periodically broadcasts the agent advertisement message to all of the BSs in the subnetwork. It is because the beacon message used in the local handoff protocol is usually much shorter than the agent advertisement message and also its sending rate is much higher than the maximum sending rate of the agent advertisement message. When we compare the Mobile IP with route optimization extension with the base Mobile IP in the next section, this method is applied to both of those protocols for fair comparison. Note that this cooperation with the local handoff is allowed in the Mobile IP standard [13].

We next consider the requirement on the buffer size to recover all the packets dropped during the handoff. We consider the case where the correspondent host

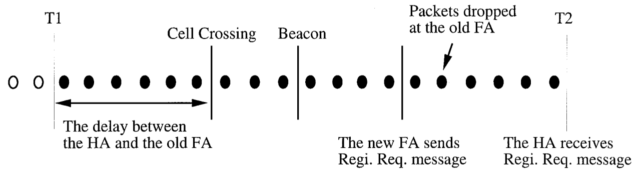

T1
Cell Crossing
Beacon
Packets dropped at the old FA
T2
The delay between the HA and the old FA
The new FA sends Regi. Req. message
The HA receives Regi. Req. message

Fig. 5 Packets dropped during the handoff in the case of base Mobile IP.

sends packets to the mobile host through the TCP connection established between them. Then, by taking account of the sending rate of TCP and the period during which in-flight packets are dropped, we can approximately determine the number of packets dropped during the handoff as follows;

$$\min\left(\frac{mws}{rtt} \times t_{loss}, mws\right) \tag{1}$$

where

mws: the possible maximum window size of the TCP connection,
rtt: the round-trip time of the TCP connection,
t_{loss}: the period during which in-flight packets are dropped.

We next consider t_{loss}; the period during which in-flight packets are dropped. It depends on the underlying handoff protocol. In the case of the base Mobile IP, as shown in Fig. 5, all the packets forwarded by the HA during the period from T1 to T2 are dropped at the old FA. Thus, we are able to determine t_{loss} for the base Mobile IP as follows;

$$t_{loss} = t_{delay} + t_{beacon} + t_{new\_fa} + t_{ha} \tag{2}$$

where

t_{delay}: the delay between the HA and the old FA,
t_{beacon}: the period from the time when the mobile host moves into the new foreign network to the time when it receives the beacon message from the new BS,
t_{new_fa}: the period from the time when the mobile host receives the beacon message to the time when the new FA sends the registration request message,
t_{ha}: the period from the time when the new FA sends the registration request message to the time when the HA receives it.

If we assume the non-overlapping cell case as the worst case, the worst case value of t_{beacon} is approximately determined as the period of beacon message. Similarly, we can derive t_{loss} for the route optimization extension as follows;

$$t_{loss} = t_{beacon} + t_{new\_fa} + t_{old\_fa} \tag{3}$$

where

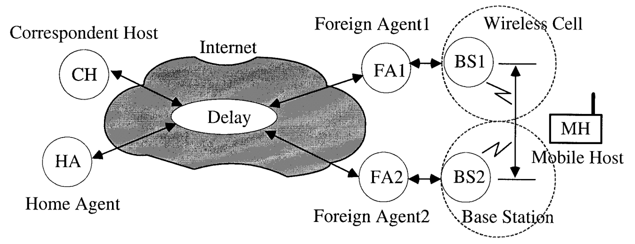


Fig. 6 Simulation model.

t_{old_fa}: the period from the time when the new FA sends the binding update message to the time when the old FA receives it.

By comparing Eqs. (2) and (3), we can find that the value of t_{loss} is much larger in the base Mobile IP case than in the route optimization extension case. Since the old FA is always placed near the new FA, the value of t_{old_fa} tends to be very small. Thus, the difference is approximately the twice of the delay between the HA and the old (new) FA.

3. Simulation Results and Discussions

In this section, we first describe our simulation model in Sect. 3.1. We then show how the smooth handoff of the Mobile IP with route optimization extension improves the performance of TCP by comparing with the case of the base Mobile IP protocol in Sect. 3.2. We next investigate the impact of packet buffering on the performance of TCP under our proposed protocol in Sect. 3.3. We finally discuss the problems of packet buffering method and possible solutions in Sect. 3.4.

3.1 Simulation Model

We consider the network configuration shown in Fig. 1 for investigating the performance of TCP in Mobile IP based networks. We assume that 2 Mbps WaveLAN is used as the wireless access network to the Internet and the BS performs as a link-layer bridge between wireless 2 Mbps WaveLAN and wired 10 Mbps Ethernet networks. Furthermore, we assume that the fixed correspondent host is connected to 10 Mbps Ethernet. For this configuration, we developed a simulation model shown in Fig. 6.

The delay taken to send a packet between FAs is set to 1 msec by considering 10 Mbps Ethernet used for the transmission. Similarly, the delay taken to send a packet between the FA and the BS is also set to 1 msec. According to the experiment results of [5], the maximum achievable throughput for the TCP connection in 2 Mbps WaveLAN is about 1.6 Mbps in the presence of no packet loss. We use this result in determining the service rate of WaveLAN, by which we can determine the delay taken to send the packet between the BS and the mobile host by considering the packet size. Other

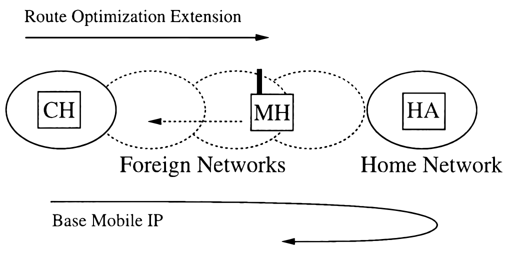


Fig. 7 Scenario to see effects of the route optimization.

delays such as the delay between the HA and the FA are dependent on the distance between the two nodes considered, which are modeled by the delay station as shown in Fig. 6.

We consider the scenario in which the fixed correspondent host transmits 1 Mbytes file to the mobile host using TCP and only one handoff event occurs during the file transfer. A congestion control algorithm adopted in TCP Tahoe is used in this scenario. The maximum segment size and the maximum window size for the TCP connection are set to 1,024 bytes and 32 segments, respectively. The resolution of the TCP timer is set to 500 msec. In our simulation experiments, we assume that there is no packet loss by network congestion or transmission error.

In [1], the authors investigated how the period of beacon message gives impacts on the number of packets dropped during the handoff by using WaveLAN. The number of packets dropped during the handoff decreases as the period of beacon message becomes shorter, but the throughput of TCP is much decreased when the period is too short. It is because the load of WaveLAN increases as the period of beacon message becomes shorter. Their results show that the TCP throughput remains high when that period is kept above 50 msec, but begins to drop significantly when that period is decreased below 50 msec. By take account of this result, we always set the period of beacon message to 50 msec.

3.2 Comparison between the Base Mobile IP and the Mobile IP with Route Optimization Extension

We first investigate how the route optimization extension of Mobile IP gives impacts on the performance of TCP. For this purpose, we consider the scenario shown in Fig. 7. The mobile host approaches toward the correspondent host along the straight line between the correspondent host and the HA, after departing from its home network. Therefore, when the route optimization extension is used, the round-trip time of the TCP connection between the correspondent host and the mobile host becomes smaller as shown in Table 1 because the correspondent host sends TCP data packets directly to the mobile host. The parameter α shown in Tables 1 and 2 corresponds to the round-trip time between the FA and the mobile host, which is roughly 8 msec. On the other hand, when the base Mobile IP is used, the round-trip time is not changed because the correspondent host sends TCP data packets to the mobile host through the HA. Note that in both cases, the mobile host sends TCP ACK packets directly to the correspondent host.

Table 1 Delays between nodes in the case of the scenario to see effects of the route optimization.

Delay (msec) between			TCP rtt (msec)	
CH & HA	HA & FA	CH & FA	Base M-IP	Route M-IP
80	0	80	160 $+\alpha$	160 $+\alpha$
80	10	70	160 $+\alpha$	140 $+\alpha$
80	20	60	160 $+\alpha$	120 $+\alpha$
80	30	50	160 $+\alpha$	100 $+\alpha$
80	40	40	160 $+\alpha$	80 $+\alpha$
80	50	30	160 $+\alpha$	60 $+\alpha$
80	60	20	160 $+\alpha$	40 $+\alpha$
80	70	10	160 $+\alpha$	20 $+\alpha$

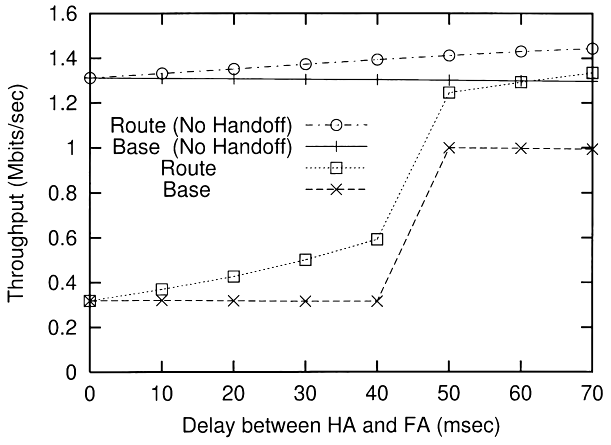


Fig. 8 Effects of the route optimization.

Figure 8 shows the throughput of TCP under our scenario. On the whole, the route optimization gives better performance than the base Mobile IP as expected. When the route optimization extension is used, the round-trip time of the TCP connection decreases as the mobile host approaches toward the correspondent host because the delay between the correspondent host and the FA decreases. As a result, the TCP throughput increases. On the other hand, when the base Mobile IP is used, the round-trip time of the TCP connection is not changed because of its inefficient routing. This is the reason why the route optimization extension gives better performance than the base Mobile IP in this scenario. Note that in both cases, the throughput of TCP increases abruptly when the delay between the HA and the FA is changed to 50 msec, although the round-trip time of the TCP connection remains constant especially in the case of the base Mobile IP. It is related to the number of packets dropped during the handoff. We

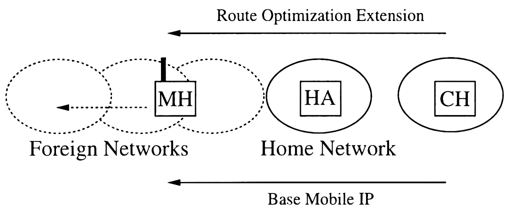


Fig. 9 Scenario to see effects of the smooth handoff.

Table 2 Delays between nodes in the case of the scenario to see effects of the smooth handoff.

Delay (msec) between			TCP rtt (msec)	
CH & HA	HA & FA	CH & FA	Base M-IP	Route M-IP
10	0	10	20 $+\alpha$	20 $+\alpha$
10	10	20	40 $+\alpha$	40 $+\alpha$
10	20	30	60 $+\alpha$	60 $+\alpha$
10	30	40	80 $+\alpha$	80 $+\alpha$
10	40	50	100 $+\alpha$	100 $+\alpha$
10	50	60	120 $+\alpha$	120 $+\alpha$
10	60	70	140 $+\alpha$	140 $+\alpha$
10	70	80	160 $+\alpha$	160 $+\alpha$

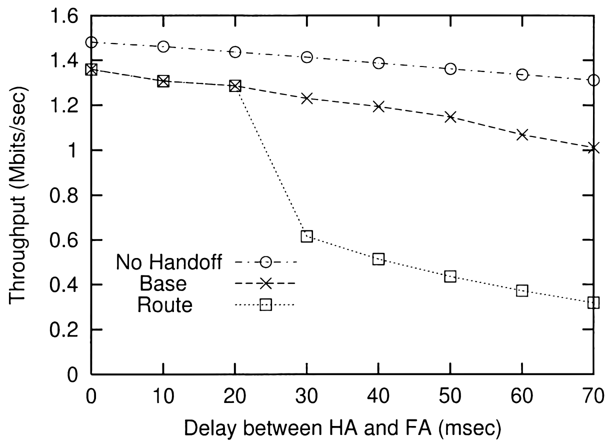


Fig. 10 Effects of the smooth handoff.

will explain its reason in more detail at the end of this subsection.

We next investigate how the smooth handoff by the route optimization extension of Mobile IP gives impacts on the performance of TCP. For this purpose, we consider the scenario shown in Fig. 9 where the mobile host becomes away from the HA and the correspondent host. In this case, the round-trip time of the TCP connection between the correspondent host and the mobile host becomes identical for two Mobile IP protocols as shown in Table 2. Therefore, we can focus on the investigation of the smooth handoff. Figure 10 shows the throughput of TCP under this scenario. On the whole, the base Mobile IP gives better performance than the route optimization extension, although the smooth handoff is supported in the route optimization extension to reduce the number of packets dropped during the hand-

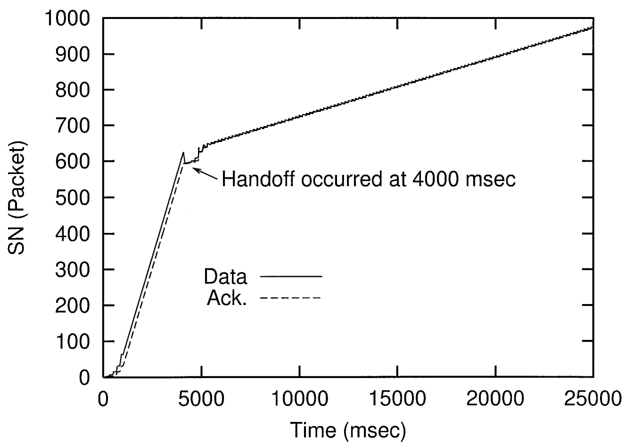


Fig. 11 TCP Tahoe behavior during the handoff.

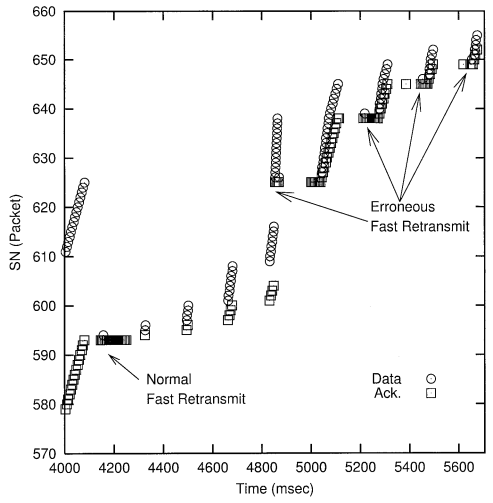


Fig. 12 Erroneous fast retransmit by multiple packet losses.

off. This result is contrary to our expectation.

To see the cause of this result, we investigate how TCP Tahoe behaves when the handoff occurs. Figure 11 shows sequence numbers of TCP data and ACK packets for the case where the delay between the HA and the FA is 70 msec, when the route optimization extension is used. In this case, the handoff occurs after the TCP sender achieves its maximum window size 32 packets, and 12 packets 594–605 are dropped from one window of data during the handoff. Other packets 606–625 forwarded by the old FA arrive at the mobile host successfully. Figure 12 shows the behavior of TCP Tahoe after the handoff in more detail. Each time the packet not dropped during the handoff arrives at the TCP receiver, it sends the corresponding ACK for packet 593. Then, the TCP sender begins slow-start upon receiving three duplicate ACKs for packet 593. After sending packets 601–608 at time around 4,660 msec, it receives the first ACK for packet 625 triggered by packet 605 at time 4,853 msec because packets 606–625 are already received by the

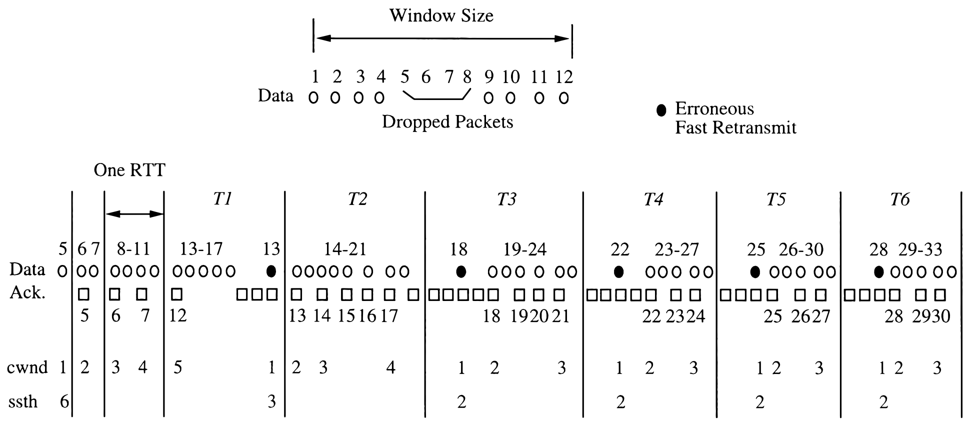


Fig. 13 Example of erroneous fast retransmit by multiple packet losses.

TCP receiver. This one large ACK forces the TCP sender to transmit 13 packets 626–638 at line speed because the congestion window is 13 packets at this point. Note that this surge of packets gives worse impacts on the network. Then, the TCP sender transmits packet 626 at time 4,870 msec upon receiving three duplicate ACKs for packet 625, which are triggered by packets 606–608. But, this is an erroneous Fast Retransmit because it is not related to packet loss. After time around 5,200 msec, an erroneous Fast Retransmit occurs every round-trip time until the file transfer is completed, and every packet is transmitted twice, and in most of time no more than 6 packets are transmitted during one round-trip time.

We describe the reason why such erroneous Fast Retransmit occurs consecutively in Fig. 13. We assume that packets 5–8 are dropped from one window of data during the handoff and other packets 9–12 are received successfully by the TCP receiver. In this case, the TCP sender retransmits packet 5 upon receiving three duplicate ACKs for packet 4. After two round-trip times (i.e., in the period $T1$), the TCP sender receives four ACKs for packet 12 triggered by packets 8–11. When receiving the first ACK for packet 12, the TCP sender transmits packets 13–17 as response because the congestion window is 5 at this point. Then, it begins slow-start to recover packet 13 upon receiving three duplicate ACKs for packet 12, although packets 13–17 are already transmitted just before. As a result of that, after one round-trip time, the TCP sender receives five ACKs instead of one ACK, it thus transmits eight packets 14–21 unlike the case of normal Fast Retransmit in which just two packets are expected to be transmitted. Note that packets 14–17 are transmitted again because the TCP sender in slow-start ignores the fact that those packets were transmitted before. For this reason, an erroneous Fast Retransmit occurs after one round-trip time, and that in turn triggers another erroneous Fast Retransmit after one round-trip time. In this way, an erroneous Fast Retransmit occurs every round-trip time

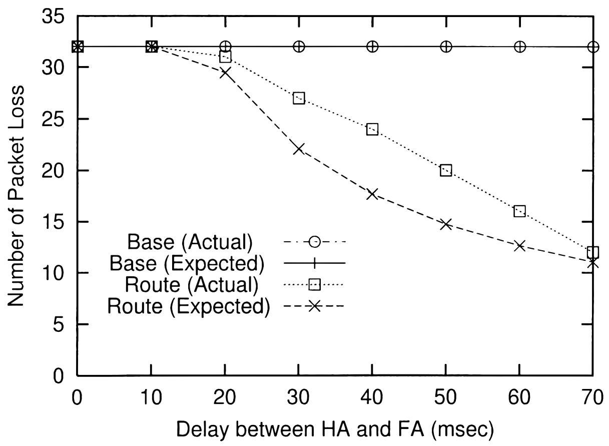


Fig. 14 Relation between number of packet losses and round-trip time.

until the file transfer is completed. After the first erroneous Fast Retransmit, the number of packets transmitted during one round-trip time becomes smaller than before until it becomes 6, and then it is never changed. Also, we can see that more packets than the congestion window size are transmitted during one round-trip time when an erroneous Fast Retransmit occurs. Such erroneous Fast Retransmit occurs consecutively if the following two conditions are satisfied simultaneously;

- more than three packets are dropped consecutively during the handoff,
- more than two packets successfully arrive at the TCP receiver after the handoff.

Unfortunately, those conditions are likely to be satisfied in the case of the route optimization extension. That is the reason why the performance of TCP is degraded further when the route optimization extension is used. We next investigate the number of packets dropped during the handoff which is closely related to the performance of TCP.

Figures 14 and 15 show the number of packets dropped and the packet loss period t_{loss} for the case

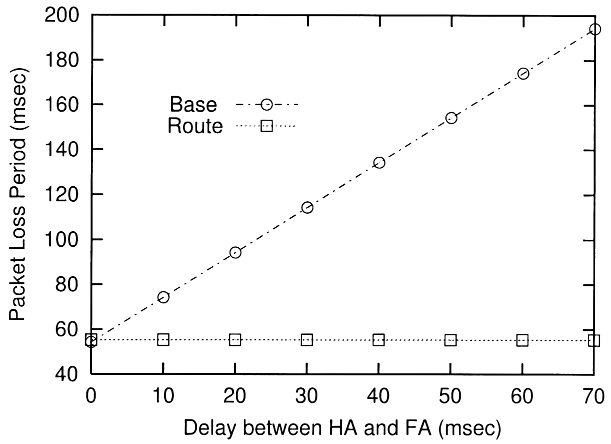


Fig. 15 Relation between packet loss period and round-trip time.

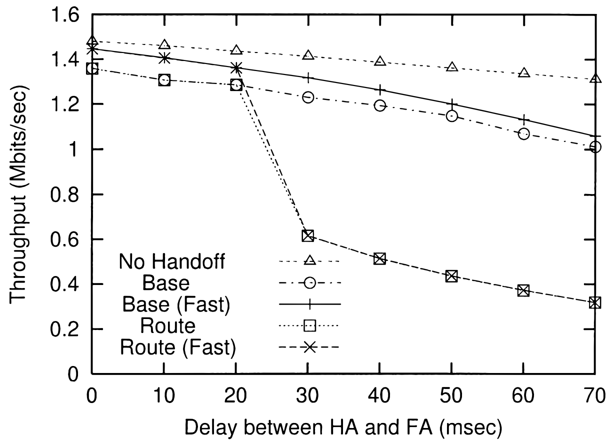


Fig. 16 Comparison with the fast retransmit method.

of Fig. 10, respectively. As shown in Fig. 15, t_{loss} is proportional to the round-trip time in the case of the base Mobile IP due to its inefficient routing. In this case, as we can see from Eqs. (1) and (2), most of packets are always probably dropped from one window of data without regard to the round-trip time, especially when the correspondent host is placed close to the HA. For this reason, 32 packets are always dropped without regard to the round-trip time in the case of the base Mobile IP. Note that the actual number of packets dropped during the handoff can be larger than the theoretical value. It is because packets arrive at the BS in a burst manner although we assumed implicitly in Eq. (1) that the intervals between two packets are equal.

On the other hand, in the case of the route optimization extension, t_{loss} is almost constant without regard to the round-trip time and t_{beacon} becomes a dominant factor in t_{loss} because the old FA is always placed near the new FA. For this reason, when the round-trip time becomes larger than t_{beacon}, the conditions for triggering consecutive erroneous Fast Retransmit are satisfied in the case of the route optimization extension. In the case of the base Mobile IP, however, the TCP sender always restarts the file transfer after the retransmission timeout and no erroneous Fast Retransmit occurs because all of the packets within one window of data are dropped so that the TCP receiver can send no duplicate ACK. That is the reason why the base Mobile IP gives better performance than the route optimization extension under the scenario to see effects of the smooth handoff, as shown in Fig. 10.

From the above results, we can see that the smooth handoff of the route optimization extension is useful only if the round-trip time is much larger than t_{loss} so that no more than three packets are dropped. In that case, the route optimization extension gives better performance than the base Mobile IP because all of the dropped packets are recovered by normal Fast Retransmit and no erroneous Fast Retransmit occurs. Otherwise, when the route optimization extension is used, the performance of TCP can be further degraded compared to the case of the base Mobile IP due to its smooth handoff. Note that even when the round-trip time is much larger than t_{loss}, there is a possibility that more than three packets can be dropped because packets arrive at the BS in a burst manner. On the other hand, in Fig. 8, we showed that the route optimization extension gives better performance than the base Mobile IP under the scenario to see effects of the route optimization. However, as mentioned before, it is because in that scenario the round-trip time of TCP becomes smaller due to the route optimization. In that scenario, the smooth handoff of the route optimization extension just gives an ill effect on the performance of TCP like in Fig. 10, especially when the delay between the HA and the FA is less than 50 msec. We next explain the reason why the throughput of TCP increases abruptly when the delay between the HA and the FA is changed to 50 msec for both cases, which is deferred. It is because most of packets begin to be dropped from one window of data from the point that the delay between the HA and the FA is 50 msec so that no erroneous Fast Retransmit occurs, whereas the conditions for triggering consecutive erroneous Fast Retransmit are satisfied when the delay is less than 50 msec, in both cases.

3.3 Impacts of Buffering in Mobile IP Based Networks

In this subsection, before investigating the buffering method, we first show how the performance of TCP is affected by the fast retransmit method [2]. In doing so, we use the parameters used in obtaining Fig. 10, and thus the number of packets dropped during the handoff is equal to the case of Fig. 10. When the fast retransmit method is used, the mobile host sends three duplicate ACKs artificially upon receiving the ACK for the handoff request message from the new BS. Figure 16 shows the results. In the case of the base Mobile IP,

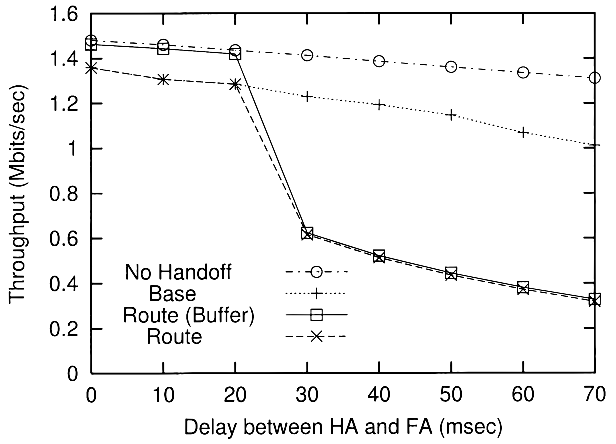


Fig. 17 Effects of the buffering method.

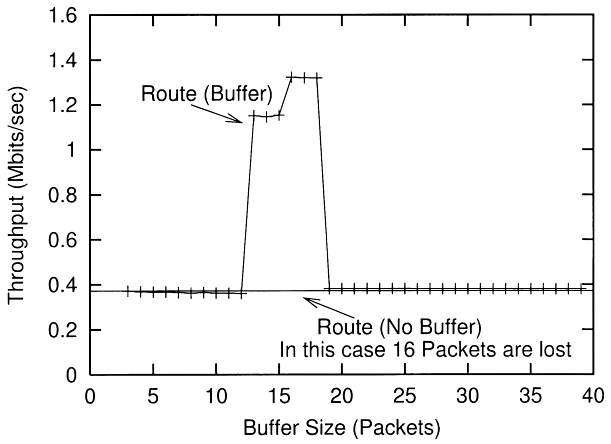


Fig. 18 Relation between TCP throughput and buffer size.

the performance of TCP is always improved without regard to the round-trip time if the Fast Retransmit method is supported because the method makes the TCP sender restart the file transfer without waiting for the retransmission timeout and erroneous Fast Retransmit, when all of the packets within one window of data are dropped. For the same reason, if the fast retransmit method is incorporated with the route optimization extension, the performance of TCP is improved when all of the packets within one window of data are dropped. From the results, we can see that this method is useful only if most of packets are dropped from one window of data during the handoff or the window size is too small, so that the number of packets arriving at the mobile host after the handoff is too small to generate three duplicate ACKs. Otherwise, it gives very little impacts on the performance of TCP because the time to start Fast Retransmit is not much different whether it is supported or not.

Figure 17 shows the impacts of the buffering method on the performance of TCP when using the proposed protocol which makes buffering at a BS possible. In this figure, we used the buffer of 32 packets, which corresponds to the maximum window size. Therefore, we can recover all the packets dropped during the handoff because the number of packets dropped cannot exceed the maximum window size. Other parameters are same to Fig. 10. We can see that the performance of TCP is not improved by the buffering method even when the size of buffer is larger than the number of packets dropped. It is because erroneous Fast Retransmit occurs consecutively if the old BS sends more than two packets which are already received by the mobile host toward the new BS after the handoff. This situation is likely to be happened because BSs cannot know which packets are dropped.

Let us consider the impacts of the buffer size on the performance of TCP in more detail. We assume that buffers behave like FIFO (First In First Out) queue. That is, when a packet arrives at the buffer with no space, it is stored in the tail of the buffer and instead the packet arrived at first is dropped from the head of buffer. Therefore, if the buffer size is smaller than the number of packets dropped, some packets dropped earlier cannot be recovered. On the other hand, if the buffer size is larger than the number of packets dropped, the old BS sends some packets which are already arrived at the mobile host successfully, toward the new BS after the handoff, and thus those packets trigger duplicate ACKs at the mobile host.

Figure 18 shows the impacts of the buffer size on the performance of TCP when 16 packets are dropped during the handoff. We can see that the range of the buffer size improving the throughput of TCP is narrow. When the difference between the buffer size and the number of packets dropped is lager than two, the old BS sends more than two packets already received by the mobile host before the handoff, toward the new BS, and thus the TCP receiver sends more than two duplicate ACKs. Then, unfortunately, erroneous Fast Retransmit occurs consecutively at the TCP sender like the case that multiple packets are dropped during the handoff. It is a critical problem that the throughput of TCP is not improved by the buffering method even when the buffer size is larger than the number of packets dropped. It is because we must provide a large buffer to be able to recover all of the packets dropped during the handoff at any time. In the next subsection, we will discuss the problems of the buffering method in more detail.

3.4 Problems of the Buffering Method

In the previous subsection, we have shown that the performance of TCP cannot be improved without an appropriate buffer size even when the buffer size is larger than the number of packets dropped during the handoff. This is one of problems that should be addressed to make the buffering method feasible because we cannot estimate exactly the number of packets dropped during the handoff even if we can know the round-trip time, the maximum window size, and the packet loss period. To address this problem, it is necessary to satisfy following two conditions; (1) the buffer size must be larger than the number of packets dropped during the handoff and (2) the old BS must not send more than two packets already received by the mobile host before the handoff, toward the new BS. We first consider the first condition. The buffer size can be determined as the size of the transmission buffer which is implemented in the BS to sends packets toward the mobile host. The congestion window size never gets larger than the size of the transmission buffer even if the wireless link is not the bottleneck link of the TCP connection. Thus, the number of packets dropped is always less than or equal to the size of the transmission buffer. The second condition can be satisfied if we manage the buffer for recovering packet loss by using the ACK of TCP or the ACK of the link-layer protocol used in the wireless link. In that case, the old BS can drop the acknowledged packets from the buffer, and thus the mobile host never receives duplicate packets generated by introducing the buffering method.

4. Conclusions

We have shown that in most cases, the smooth handoff by the route optimization extension of current Mobile IP standard cannot prevent the degradation of TCP performance due to handoffs, although it is designed to reduce the number of packets dropped during the handoff. Moreover, we have found that in the case of the route optimization extension, the performance of TCP can be further degraded than the case of the base Mobile IP unless its smooth handoff makes less than four packets be dropped during the handoff. To address such problem, we have proposed a buffering method in which only one base station performs buffering in order to recover the packets dropped during the handoff without the scalability problem. In doing that, we have modified the route optimization extension in order to support buffering of packets at a base station. Finally, we have discussed the problems that should be addressed to recover the packets dropped during the handoff by the buffering method without giving a worse impact on the performance of TCP, and proposed our solution to resolve those problems.

In this paper, we have not considered other TCP versions such as TCP Reno and TCP New Reno. If packet buffering is not supported, the performance of TCP would become different depending on the TCP version. It is because each TCP version employs its own congestion control algorithm. That is, when packet losses occur due to the handoff, each TCP version reacts differently on the pattern of packet losses and the number of packet losses according to its congestion control algorithm. However, if the packets dropped during the handoff are successfully recovered by the proposed buffering method, the handoff event becomes transparent to the upper layer protocols than Mobile IP. Thus, there is no performance difference between TCP versions as far as we are only concerned with handoff. It means that the buffering method does not depend on the TCP version. It is one major advantage of the buffering method. Note that if the buffering method is successfully employed, the upper layer protocols having been designed for wired networks can be used on the mobile hosts without any modification.

References

[1] R. Caceres and V. Padmanabhan, "Fast and scalable handoffs for wireless networks," Proc. ACM Mobicom'96, pp.56–66, Nov. 1996.

[2] R. Caceres and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," IEEE J. Sel. Areas Commun., vol.13, no.5, pp.100–109, Nov. 1995.

[3] K. Brown and S. Singh, "M-TCP: TCP for mobile cellular networks," ACM Computer Communications Review, vol.27, no.5, pp.19–43, Oct. 1997.

[4] S. Seshan, H. Balakrishnan, and R.H. Katz, "Handoffs in cellular wireless networks: The daedalus implementation and experience," Wireless Personal Communications, vol.4, no.2, pp.141–162, March 1997.

[5] H. Balakrishnan, S. Seshan, and R.H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," Wireless Networks, vol.1, no.4, pp.469–481, Dec. 1995.

[6] P. Manzoni, D. Ghosal, and G. Serazzi, "Simulation study of the impact of mobility on TCP/IP," Proc. International Conference on Network Protocols, pp.196–203, 1994.

[7] A. Bakre and B.R. Badrinath, "I-TCP: Indirect TCP for mobile hosts," Proc. 15th International Conference on Distributed Computing Systems, pp.136–143, June 1995.

[8] R. Yavaatkar and N. Bhagwat, "Improving end-to-end performance of TCP over mobile internetworks," Proc. Workshop on Mobile Computing and Applications, pp.146–152, Dec. 1994.

[9] A. Bakre and B.R. Badrinath, "Handoff and systems support for indirect TCP/IP," 2nd Usenix Symposium on Mobile and Location-Independent Computing, April 1995.

[10] K.Y. Wang and S.K. Tripathi, "Mobile-end transport protocol: An alternative to TCP/IP over wireless links," Proc. INFOCOM, vol.3, pp.1046–1053, 1998.

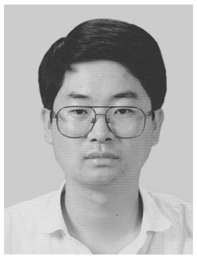
[11] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, and R.H. Katz, "A comparison of mechanisms for improving tcp performance over wireless links," IEEE/ACM Trans. Networking, vol.5, no.6, pp.756–769, Dec. 1997.

[12] C.E. Perkins, "Mobile IP," International Journal of Communication Systems, pp.3–20, 1998.

[13] C.E. Perkins, "IP mobility support version 2," draft-ietf-mobileip-v2-00.txt, 1997.

[14] C.E. Perkins, "Mobile IP," IEEE Commun. Mag., pp.84–99, May 1997.

[15] C.E. Perkins and D.E. Johnson, "Route optimization in Mobile-IP," draft-ietf-mobileip-optim-07.txt (work in progress), 1997.



Doo Seop Eom received the B.E. and M.E. degrees in Electronics Engineering from Korea University, Seoul, Korea, in 1987 and 1989, respectively. He joined the Communication Systems Division, Electronics and Telecommunications Research Institute (ETRI), Korea, in 1989. He received the Dr.E. degree in Information and Computer Sciences from Osaka University, Osaka, Japan, in 1999. He was awarded a Japanese Government Scholarship during 1996–1999 his academic years at Osaka University. From September 1999 to August 2000, he was an Associate Professor of Wonkwang University, Korea. Since September 2000, he has been an Assistant Professor in the School of Electrical Engineering, Korea University, Seoul, Korea. His research interests include communication network design and wireless ATM.

Masashi Sugano received the B.E., M.E., and D.E. degrees in Information and Computer Sciences from Osaka University, Japan, in 1986, 1988, and 1993, respectively. In 1988, he joined Mita Industrial Co. Ltd. (currently, Kyocera Mita Corporation) as a Researcher. From September 1996, he has been an Associate Professor of Osaka Prefecture College of Health Sciences. His research interests include design and performance evaluation of computer communication networks, network reliability, and wireless network systems. He is a member of IEEE, ACM and IEICE.

Masayuki Murata received the M.E. and D.E. degrees in Information and Computer Sciences from Osaka University, Japan, in 1984 and 1988, respectively. In April 1984, he joined Tokyo Research Laboratory, IBM Japan, as a Researcher. From September 1987 to January 1989, he was an Assistant Professor with Computation Center, Osaka University. In February 1989, he moved to the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. From 1992 to 1999, he was an Associate Professor in the Graduate School of Engineering Science, Osaka University, and from April 1999, he has been a Professor of Osaka University. He moved to Advanced Networked Environment Division, Cybermedia Center, Osaka University in April 2000. He has more than two hundred papers of international and domestic journals and conferences. His research interests include computer communication networks, performance modeling and evaluation. He is a member of IEEE, ACM, The Internet Society, and IPSJ.

Hideo Miyahara received the M.E. and D.E. degrees from Osaka University, Osaka, Japan in 1969 and 1973, respectively. From 1973 to 1980, he was an Assistant Professor in the Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, Kyoto, Japan. From 1980 to 1986, he was an Associate Professor in the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University, Osaka, Japan. From 1986 to 1989, he was a Professor of the Computation Center, Osaka University. Since 1989, he has been a Professor in the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. From 1983 to 1984, he was a Visiting Scientist at IBM Thomas J. Watson Research Center. His research interests include performance evaluation of computer communication networks, broadband ISDN, and multimedia systems. He is a fellow of the IEEE.