Enhancing TCP Performance over Wireless Network with Variable Segment Size

Keuntae Park, Sangho Park, and Daeyeon Park

Abstract: **TCP, which was developed on the basis of wired links, supposes that packet losses are caused by network congestion. In a wireless network, however, packet losses due to data corruption occur frequently. Since TCP does not distinguish loss types, it applies its congestion control mechanism to non-congestion losses as well as congestion losses. As a result, the throughput of TCP is degraded. To solve this problem of TCP over wireless links, previous researches, such as split-connection and end-to-end schemes, tried to distinguish the loss types and applied the congestion control to only congestion losses; yet they do nothing for non-congestion losses.**

We propose a novel transport protocol for wireless networks. The protocol called VS-TCP (Variable Segment size Transmission Control Protocol) has a reaction mechanism for a non-congestion loss. VS-TCP varies a segment size according to a non-congestion loss rate, and therefore enhances the performance. If packet losses due to data corruption occur frequently, VS-TCP decreases a segment size in order to reduce both the retransmission overhead and packet corruption probability. If packets are rarely lost, it increases the size so as to lower the header overhead. Via simulations, we compared VS-TCP and other schemes. Our results show that the segment-size variation mechanism of VS-TCP achieves a substantial performance enhancement.

Index Terms: **Wireless networks, TCP, variable segment size, end-to-end approach.**

I. INTRODUCTION

An interest in the Internet has been exponentially growing owing to the exploding popularity of World Wide Web(WWW) and the wide spread of communication equipments, such as Ethernet, ISDN, ATM, and ADSL. As newly developed network technologies are introduced continuously, fresh technologies and previously deployed ones coexist in the Internet. These heterogeneous networks can work as a global network by means of Transmission Control Protocol(TCP) [1] that operates without respect to underlying hardwares. This hardware independence of TCP makes it a dominant protocol in the Internet. A new trend in the Internet is the appearance of wireless networks, such as CDPD, GSM and Wireless LAN. The remarkable characteristics of wireless networks, i.e., host mobility and portability, accelerate the deployment of the wireless networks. However, since TCP has been developed on the assumption of wired link, TCP over wireless link has following problems.

A packet loss can be classified into two types: a congestion loss and a non-congestion loss. If a queue of any intermediate routers between sender and receiver becomes full owing to excessive network traffic, the router drops additional received packets. This kind of packets drop is called congestion loss. A non-congestion loss comes from data corruption due to the noise of channel. While a non-congestion loss rarely happens in a wired network, the channel of wireless network is very unreliable and hence the wireless network suffers considerable non-congestion losses.

When a packet is lost, TCP always retransmits the packet in order to deliver data reliably. As TCP is designed for a wired network, it assumes all the losses are caused by congestion. Since retransmitted packet may aggravate network congestion, the congestion control mechanism of TCP reduces data sending rate aggressively. However, the wireless network essentially has non-congestion losses and TCP applies the congestion control even for a non-congestion loss. Therefore, the throughput of TCP can be degraded significantly in wireless network because of the congestion control.

To solve this problem in a wireless network, previous researches proposed several transport protocols, such as I-TCP [2], METP [3], WTCP [4], and Freeze-TCP [5]. I-TCP and METP split a TCP connection at a *base station* that locates on the boundary between wireless and wired networks. Since a wireless link is typically the last one hop of the entire connection, no congestion loss exists in the wireless TCP connection. Consequently, in the *split-connection schemes*, two types of packet losses are perfectly separated since congestion losses occur only in a wired connection and non-congestion ones mostly do in a wireless connection. However, the schemes induce large overhead in the base station, which splits a end-to-end connection and manages both connections. To reduce this overhead, the *end-to-end schemes*, such as WTCP and Freeze-TCP, were suggested. They preserve one TCP connection between two end-to-end nodes and estimate the loss type on the basis of the subsidiary informations, e.g., transmission error pattern or signal power. However, they have the problem of incorrectness in discriminating the loss type.

Both approaches tried to distinguish the loss type, and apply the congestion control mechanism to congestion losses only. However, they do nothing for the non-congestion losses even though the rate of non-congestion loss varies with time. Since TCP is optimized for the long-term average of the non-congestion loss rate, they are not adequate in a short-term view. Our proposal starts from this point and we try to improve TCP performance by properly reacting to the non-congestion loss.

Manuscript received March 23, 2001; approved for publication by Dong In Kim, Division II Editor, March 18, 2002.

K. Park and S. Park are with the Electrical Engineering and Computer Science Department, KAIST, Korea, e-mail: {ktpark, shpark}@sslab.kaist.ac.kr.

D. Park is with the Faculty of Electrical Engineering and Computer Science, KAIST, Korea, e-mail: daeyeon@ee.kaist.ac.kr.

In this paper, we propose a novel transport protocol, *Variable Segment size Transmission Control Protocol* (VS-TCP), which provides a reaction mechanism against the non-congestion loss. VS-TCP varies the size of segment according to a non-congestion loss rate; *segment*, which is composed of TCP header and data, is the transmission unit of TCP. In a typical TCP connection, the maximum segment size is determined on connection establishment and remains fixed during the connection. However, VS-TCP dynamically adjusts the maximum segment size of a connection according to a current non-congestion loss rate. When packets are frequently lost, VS-TCP decreases the maximum segment size in order to reduce the overhead of retransmission and the probability of packet corruption. If the packet loss rarely occurs, it increases the maximum segment size and reduces the header overhead. The segment-size control mechanism should be applied to non-congestion loss only. For the loss-type discrimination, our scheme divides a connection like the split-connection schemes do. While the original TCP is deployed in a wired connection, the non-congestion loss control mechanism is applied only to a wireless connection. Therefore, congestion losses are handled by the congestion control of the original TCP and non-congestion losses are efficiently handled by VS-TCP.

The rest of this paper is organized as followings. A detailed background is presented in Section II and related works are found in Section III. In Section IV, we explain the overall structure of our scheme and, especially, the segment-size variation mechanism of VS-TCP. Section V presents simulation environment and evaluation results and Section VI concludes this paper.

II. BACKGROUND

A. The Characteristics of Wireless Link

There are two obviously different characteristics between a wireless link and a wired one.

- The inherently high bit error rate
  While the packet error rate of wired LAN is usually about 10^{-8}, the recommended error rate of wireless LAN is 4×10^{-5}. In the case of wireless WAN (e.g., GSM and CDPD), moreover, the packet error rate is much higher than that of wireless LAN. For example, the packet error rate of CDPD increases from 0% to 10% as the mobile node speeds up from 0 mph to 50 mph. [4]
- The large variation of error rate
  Since the wireless link is vulnerable to interferences, the error rate varies according to circumstances. The previous study showed that the maximum variation of frame error rate is 0.28 per minute in a wireless link. [6] In a wireless network, a fixed error rate can not be assumed owing to the large variation of error rate.

B. Problems of TCP over a Wireless Link

To transmit a packet reliably, TCP retransmits a lost packet; the loss of packet is determined by whether an acknowledgment packet (ACK) is received successfully or not. After TCP sends a data packet, it waits for an ACK from the destination node. If it does not receive an ACK during a specified period of time, it regards the packet as lost and retransmits. The waiting period is calculated based on a *round trip time*, an interval from the transmission of a packet to the reception of a corresponding ACK. Until it receives an ACK, it repeatedly retransmits the missing packet and hence achieves a reliable transmission. Reference [7] gives more detailed description about data transmission mechanism of TCP. TCP was made on the basis of a wired link and, as a result, TCP over wireless links has problems. The operations that cause problems on the wireless are the congestion control mechanism and the segment size determination of TCP. Following subsections describe the TCP operations and the problems.

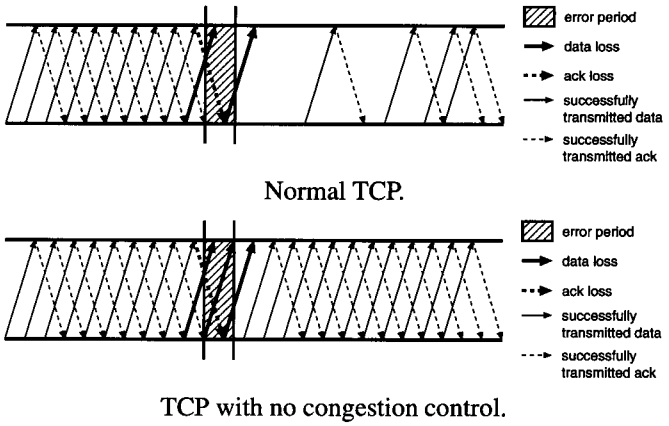


Fig. 1. The negative effect of wrongly invoked congestion control.

B.1 Congestion Control

As traffic increases, queues in routers are occupied gradually and become full finally. Then, the routers probably discard additional incoming packets. Such packet losses are called congestion losses. To dissolve a network congestion, the *congestion control mechanism* of TCP shrinks *window*, the maximum amount of data that TCP can send without waiting ACKs of previously sent data. As a result, the additional, incoming flow to the network is reduced and the congested routers may make the free space in their queues. The diminished window is restored gradually whenever TCP receives an ACK. Since the successful receipt of ACK implies the improved network status, TCP increases the rate of sending data.

Data corruption as well as network congestion can provoke a packet loss, and yet TCP does not distinguish the loss type. It regards all packet losses as caused by congestion. Even when a non-congestion loss occurs, TCP reduces the sending rate but a network may not be congested. Owing to extremely low bit error rate of wired media, such absurd behavior of TCP does not matter in wired networks. In wireless networks, however, non-congestion losses occur frequently and throughput of TCP can be greatly degraded. Fig. 1 shows an example of TCP performance degradation due to wrongly invoked congestion control. We have compared normal TCP with the modified TCP in which congestion control mechanism is removed. In the figure, after wrongly invoked congestion control, normal TCP spends long time on recovering the transmission rate. In this case, the

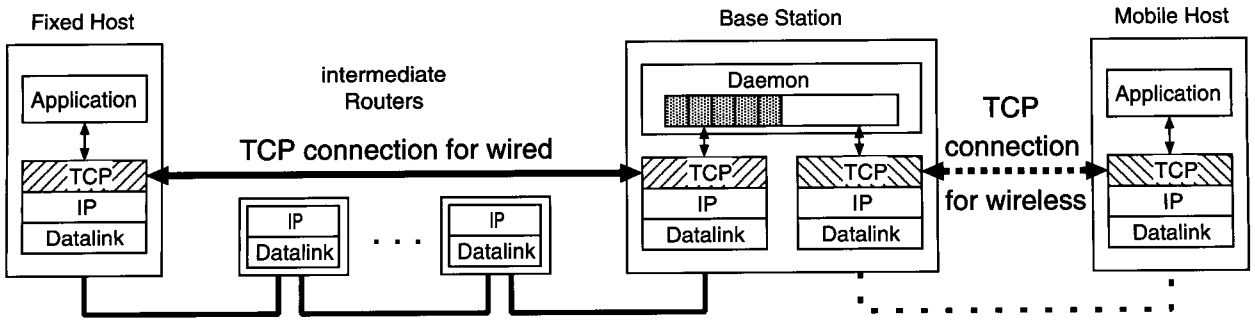


Fig. 2. Split-connection scheme.

modified TCP works better rather than normal TCP.

B.2 Segment Size of TCP

The term *segment* means the data transmission unit of TCP. A segment may be sent as one packet. Maximum segment size is determined at TCP connection establishment and is maintained until the connection closes. TCP calculates the largest segment size by means of Path-MTU discovery or uses a preset default value, in general, 512 bytes. Maximum Transmission Unit(MTU) means maximal frame size of underlying link and, if the size of packet exceeds that value, IP divides the packet into several smaller packets. Such action is called *IP segmentation*. IP segmentation is an expensive process and can degrade end-to-end performance. [8] It is avoided by using Path-MTU discovery that finds the minimum value among MTUs of links along a TCP connection. As a result, TCP acquires the largest segment size that is small enough not to incur IP segmentation. Since data transmission with larger size can reduce the header overhead, TCP sends data with the size obtained by Path-MTU discovery.

In wireless networks, however, the optimal frame size in a link-layer changes according as link conditions, such as bit error rate and latency, vary with time. Consequently, the optimal segment size of TCP connection may vary also and throughput with minimum MTU acquired by Path-MTU discovery may be lower than that with other size. The optimal segment size is not fixed in wireless networks. To achieve better performance, TCP should adapt maximum segment size for the current link conditions.

III. RELATED WORK

A. *Existing TCP Solutions*

A.1 Split-connection Scheme

As mentioned above, TCP was designed on the basis of wired link and a TCP connection over heterogeneous links therefore causes some problems. To dissolve these problems, split-connection schemes divide a TCP connection between a fixed host (FH) and a mobile host (MH) into two TCP connections. Each connection is then laid over homogeneous links. Fig. 2 gives a brief description of split-connection scheme. The base station (BS) that supports MH's mobility is the boundary node between wired and wireless networks. While the connection between FH and BS is composed of wired links only, the one between BS and MH consists of just wireless links. In the data transmission from FH to BS, BS buffers the data sent by FH and replies an ACK to FH immediately. Then, the responsibility of delivering the data reliably to MH is handed to BS. In the case of data from MH, BS also buffers the data and tosses them to FH.

The split-connection scheme has two advantages, faster retransmission and exact discrimination of loss-type. In normal TCP, lost packets are retransmitted by FH. With splitting, however, BS retransmits the packets lost on wireless links. Since the connection between BS and MH is typically one hop and is much shorter than the entire pass, the interval between loss and retransmission is much shorter than that of normal TCP, thus the retransmission is much faster than normal TCP and, in addition, the decreased window size is restored also more quickly.

The other advantage is that packet loss can be exactly distinguished if a wireless link is the last one hop of connection. Congestion losses are occurred by packet drop at an intermediate router or destination host, whose receiving buffer becomes full. Particularly in the case of one-hop connection, the destination host may not drop packets because the amount of data in transmission is limited to the free buffer size advertised by the receiver, called *advertised window*. In addition, the connection between BS and MH has no intermediate routers and therefore does not suffer from congestion losses. In the connection between FH and BS, reversely, non-congestion losses induced by data corruption rarely happen because the underlying wired links are extremely reliable. Hence, it is a reasonable assumption that the connection over wired links has congestion losses only and the one over a wireless link has non-congestion losses only. The type of packet loss can be determined by the connection where the packet was lost.

However, the split-connection scheme overburdens BS, which should manage a couple of connections and buffer, and hence BS can be a bottleneck if too many connections pass it. Moreover, an intra-area movement of MH incurs a complicate problem. MH can move to the area served by other BS and this operation of changing BS is called a *hand-off*. At a hand-off, the previous BS must hand over buffered data and state information of connections to a new BS. If the state information is not migrated completely, new connection-split at the new BS must be built again in order to set parameters, such as source and destination IP addresses, source and destination port numbers, window

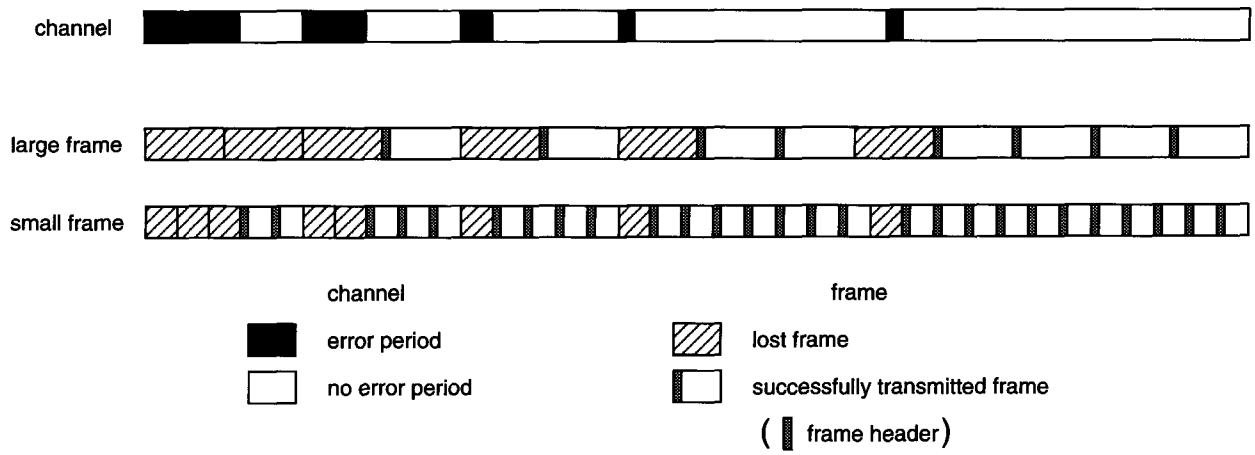


Fig. 3. The effect of frame size in variable error rate channel.

size, and sequence number. Also, data buffered at old BS must be migrated for the reliable packet delivery.

The typical example of split-connection schemes is *Indirect-TCP* (I-TCP)[2], which builds two normal TCP connections. While normal TCP works well in wired links, the TCP connection over a wireless link has a problem; it wrongly invokes the congestion control mechanism against a non-congestion loss. To dissolve this problem, other researches suggested modifying the transport-layer protocol of a wireless part, i.e., between BS and MH, which can be easily deployed and altered. *Mobile End Transport Protocol* (METP) [3], similar to ON-OFF protocol [9], adopts a modified TCP in the wireless connection. METP assumes that only the last one hop is a wireless link and link-layer retransmission covers all non-congestion losses. Hence, most functions of the original TCP are cut away and the modified TCP operates only the flow control. METP eliminates the problem of congestion control wrongly invoked but does not dissolve the problems of wired connection overhead and buffer overhead of BS.

A.2 End-to-end Scheme

End-to-end schemes distinguish loss types on the basis of end-to-end approaches. They maintain only one seamless connection between FH and MH so as to eliminate the overhead of BS.

WTCP [4] maintains the pattern of non-congestion losses and estimates non-congestion loss rate from the pattern. If current loss rate is much higher than the estimated loss rate, it is deduced that losses are caused by network congestion. This approach is based on the assumption that congestion losses are less frequent than non-congestion losses. It checks the inter-arrival time of packets and regards the increase of packet inter-arrival time as the signal of future congestion. When it detects future congestion, it reduces the data sending rate in order to avoid congestion. As a result, congestion losses become infrequent. It assumes that congestion losses are less frequent than non-congestion losses. In the real world, however, the non-congestion loss rate varies widely and it is difficult to define a fixed non-congestion loss pattern. WTCP may sometimes mistake a non-congestion loss for a congestion loss.

Freeze-TCP [5] predicts a future non-congestion loss by monitoring the physical signal strength of mobile device. It regards signal fading as an indication of future non-congestion loss. However, Freeze-TCP needs link-layer support of measuring the signal strength and depends on the signal fading pattern. In other words, it works well only when the signal strength weakens gradually.

The end-to-end schemes have no hand-off overhead and no state maintenance. However, since their loss-type predictions depend on uncertain previous informations, their loss-type discriminations are less accurate than those of the split-connection schemes.

A.3 Missing Point of Existing Solutions

The above-mentioned schemes try to apply the congestion control to only congestion losses. However, they do nothing when a non-congestion loss occurs. In this paper, we propose a novel transport protocol with a non-congestion loss control function. The proposed TCP adapts to the current wireless-channel condition that is deduced from the frequency of non-congestion loss. It invokes the proper control function for non-congestion loss and improves the performance. Our proposed mechanism focuses on not the loss-type discrimination but the 'non-congestion loss control' functionality, which the previous researches have not addressed. The novel functionality can be added to the original TCP as well as the previously mentioned protocols.

B. Varying the Frame Size in a Link-layer

Before our approach, varying the frame size in a link-layer has been already proposed. Ludwig *et al.* [6] suggested a dynamic frame size adaptation to a current link error rate. Their experiments showed that the typical frame size is too small to gain maximum performance as the header overhead is very large. To verify the idea, they increased the frame size by many times, e.g., twice and four times, and checked the performance enhancement.

Since the bit error rate of wireless link widely varies with

time, the fixed frame size cannot be optimal all the time. If the error rate is high, the small frame size has advantages since the retransmission overhead is reduced and the probability of packet error decreases. On the other hand, if the error rate is low, the large frame size is profitable as the header overhead is small. In the Fig. 3, such features are shown. Ludwig *et al.* verified that the adaptively varying frame size can be efficient in a real case by calculating the expected throughput gain, which is about 20%. They also showed that the variation of error rate is not as fast as the timeout value and hence the feedback mechanism using ACKs can be applied.

With respect to varying a size, our idea and the above scheme are similar. If a non-congestion loss is handled in a link-layer, handling packet losses of two distinct types is separated, and each layer can concentrate on one type of loss. However, we think that transport-layer handles better than link-layer owing to following reasons.

- Based on an end-to-end view: When a connection is over multiple wireless links, Ludwig *et al.*'s scheme may invoke several link-layer fragmentations but our scheme does once at a sender host. Moreover, our scheme varies size per connection but the link-layer cannot distinguish different connections. Even though the link status under each connection is different from other connections, all connections are treated as same in Ludwig *et al.*'s scheme.
- Fairness between connections: To handle non-congestion loss in a link-layer only, link-layer must reliably transfer lost frame by retransmission. As the number of frame buffer is just one, a packet toward the path in good condition may be blocked by the current retransmitting packet toward the path in bad condition. To solve this problem, link-layer must have multiple separate buffers for each connection like transport-layer as in [10].
- Hardware independent: To apply the approach suggested in [6], link-layer protocols are changed and, consequently, the underlying hardwares used within a connection should be also changed. Also, the link-layer protocol for one hardware cannot be used for another hardwares without modification. However, our scheme changes TCP that is independent of the underlying hardware. It can be applied to all kinds of wireless networks without change of hardware and the link-layer protocol.
- Hardware complexity: To vary a frame size according to a loss rate, the link-layer should additionally achieve complex operations, such as buffer management, fragmentation/assembly, and timer management. As the link-layer becomes complex, the hardware is getting more complex.

Besides, to be effective, the link-layer should manage parameters, data structures, and buffers per node. That means link-layer includes almost all TCP properties and complexities. It not only increases the system overhead by redundant operation but also contravenes the concept of layering by overlapping transport-layer with link-layer.

Compared to our solution, the link-layer approach has an advantage of wider coverage. Since ours is only applicable to TCP, other transport protocols, such as UDP, is beyond the benefits of it. On the contrary, The link-layer solution can be applied to the UDP as well. However, its retransmission mechanism increases the packet transfer time, which is harmful to UDP as UDP is best-effort service and mainly used for real-time traffic.

Table 1. Comparison between previous protocols and VS-TCP.

	normal TCP	METP	VS-TCP
segment size	fixed at connection establishment	the same as normal TCP	variable in one connection
congestion control	have	eliminated	eliminated
non-congestion loss control	none	none	added
window	minimum of advertised window, congestion window	no window, just use on-off	only advertised window

IV. VS-TCP

In this section, we explain VS-TCP, which has a reaction mechanism to non-congestion loss. We assume that a wireless link is the last one hop of entire connection. Hence, no loss due to network congestion in intermediate routers exists. Such assumption is reasonable since most of wireless environments, such as wireless LAN, CDPD, and GSM are built under the same assumption. Even without the assumption, we can check that a host is one-hop distant over a wireless link by means of an interface information and a routing table that provides the hop-count towards the host. We split a connection in order to separate non-congestion losses in a wireless link from congestion losses in wired links. From splitting, the management of congestion losses is totally shifted to normal TCP of wired part. VS-TCP is applied just to a wireless connection and handles non-congestion losses. Since we assumes no congestion loss in wireless link, VS-TCP does not handle any concerns about contention-related fairness between multiple connections over the wireless link. In the protocol, the segment size optimized to current non-congestion loss rate is dynamically changed so that the performance over the entire connection is improved.

The differences among TCP, METP, and VS-TCP are shown in Table 1. 'TCP' means normal TCP that is the wireless part of I-TCP and 'METP' is the variation of the wireless part of I-TCP. In VS-TCP, the congestion control mechanism of TCP is removed and non-congestion loss control mechanism is added. Following subsections describe these differences and VS-TCP in detail.

A. *Elimination of Unused Part of TCP*

Since no congestion loss exists in a wireless link that VS-TCP is applied, all functions related to the congestion control can be eliminated. The functions related with congestion control are activated whenever a new packet is sent, an ACK received, and any timer expired. In VS-TCP, therefore, the removal of congestion control functions reduces the processing overhead and can compensate for the additional overhead of non-congestion loss control function. Besides, the congestion window is not used and only the advertised window is used to limit the sending rate

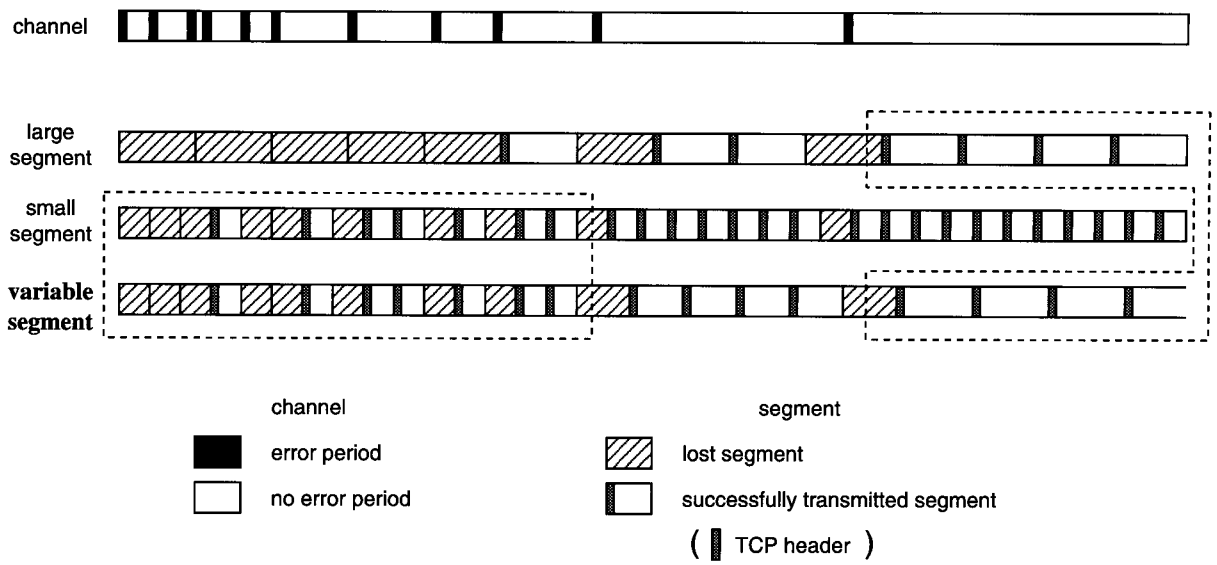


Fig. 4. The comparison among small segment, large segment and variable segment, with decreasing loss rate.

of the sender. Window size is set to the advertised window size of destination host and prevents the host's receiving buffer from overflow.

B. Non-congestion Loss Control

TCP connections over wired links are stable, and the optimal segment size of them rarely change. Hence, until the connection is closed, normal TCP deploys the segment size that is determined at the connection establishment. The segment size that may be the largest possible value without IP segmentation results in the best performance. Contrary to wired links, in a wireless network, the link status when a connection is established is different from the status when data is transmitted. Because the bit error rate varies widely with time, the link status also varies and hence the segment size determined at the connection establishment may not be adequate when data is transmitted. The key idea of non-congestion loss control mechanisms used in VS-TCP is to change a segment size adaptively according to the current non-congestion packet loss rate.

The segment-size control mechanism is based on the following observations:

- When the packet loss rate is high, the smaller segment size is more profitable owing to small retransmission overhead and small corruption probability.
- When the packet loss rate is low, a large segment size has advantages due to low header overhead. As a result, a larger segment can be sent once.

The advantages of variable segment size is illustrated in Fig. 4.

B.1 Mechanism

In this paper, we suggest two segment-size control mechanisms. The first one is similar to the congestion control mechanism of TCP and the other uses the calculated non-congestion loss rate.

- Size variation mechanism
  - Method 1

  – When a timeout occurs or a duplicate ACK is received, which means that a non-congestion loss occurs, s is decreased.

$$\text{if } s > s_{mn} \text{ then } s = \max(\frac{s}{2},\ s_{mn})$$

  – When an ACK is received, which means that data is successfully transmitted, s is increased.

$$\text{if } s < s_{mx} \text{ then } s = \min(s + \Delta s,\ s_{mx})$$

  - Method 2

  – Whenever a non-congestion loss occurs or an ACK is received, e_c and s is updated.

$$\text{if } e_c > e_w \text{and } s > s_{mn} \begin{cases} \text{if } s > \frac{s_{mx}}{2} \text{ then } s = \max(\frac{s}{2},\ s_{mn}) \\ \text{else, } s = \max(s - \Delta s,\ s_{mn}) \end{cases}$$

$$\text{else if } e_c < e_w \text{ and } s < s_{mx} \text{ then } s = \min(s+\Delta s,\ s_{mx})$$

- Used metrics
  - used in method 1, 2

  – s (Segment size) : Segment size of VS-TCP. s varies with the current error condition, in the range between s_{mn} and s_{mx}.

  – s_i (Initial segment size) : Initial value of s; the segment size at the connection establishment. This may be the same as the segment size determined by Path-MTU discovery in normal TCP.

  – Δs (Degree of change) : s is increased or decreased by the unit of Δs.

  – s_{mx}, s_{mn} (Boundary value of s) : s_{mx} is the maximum value of s and s_{mn} is the minimum.

  - used in method 2 only

  – e_w (Wanted loss rate) : Method 2 adjusts segment size to maintain current loss rate below this value.
  e_w is set when the connection is established.

- e_c (Current loss rate) :
$e_c = \frac{\text{The number of losses in previous } L_w \text{ packets}}{L_w}$
- L_w (Loss window) : L_w is the number of packets that is used to calculate e_c. If we adjust L_w to small, we can react to loss rate change more fast, however, the granularity of e_c decreases.

The mechanism of method 1 in varying a segment size is similar to that of the normal TCP in varying a congestion window. The reason why the segment size is decreased multiplicatively and increased additively is that non-congestion losses occur with burstiness like congestion losses. The burstiness of non-congestion losses means that additional losses may follow a non-congestion loss. Hence, when a loss is detected, VS-TCP assumes that additional losses will occur and reduces a segment size by half. Since VS-TCP quickly reacts against a loss, the retransmission overhead can be reduced largely. After a burst data corruption, VS-TCP increases a segment size additively; the slow increasing is better since the header overhead is lighter than the retransmission overhead. The parameters, e.g., s_{mx} and s_{mn}, influence the throughput of VS-TCP. The larger s_{mx} may increases both packet error rate and efficiency due to low header overhead. The parameter s_{mn} should be carefully selected since too high s_{mn} decreases the effectiveness of VS-TCP and the smaller value increases the header overhead.

Method 2 uses packet loss rate in varying a segment size. When the measured loss rate e_c is lower than a given rate e_w, VS-TCP increases the segment size, and, in the reverse case, it decreases. The smaller e_w decreases the average segment size but increases the header overhead. When the parameter is large, the retransmission overhead is increased. The best e_w is changed according to a real loss rate and should be selected carefully. We recommend e_w higher than the real loss rate because e_c is calculated in a short term.

In method 2, the value of loss window size L_w is closely related to the performance. If the size of L_w is large, the method can estimate current loss rate more accurately. However, its reaction time is much larger since it takes some period of time for the calculated loss rate to reflect the real loss-status. If the variation of loss rate is relatively large, method 2 may fail to follow up the real loss rate. On the contrary, if the size of L_w is small, the method can track the real loss rate quickly. However, the resolution of the calculated loss rate becomes worse so that the method sometimes mistakes just a single bit error for a signal of burst data corruption.

Compared to method 1, method 2 can be more accurate when the variation of loss rate is relatively slow since the method 2 is based on the current loss rate and can separate the intermittent loss from the burst loss. However, if loss rate varies quickly, L_w of method 2 cannot be large because of the long convergence time as mentioned before. With small L_w, the accuracy of method 2 may degrades with sparse resolution of the loss rate calculation.

In Section V, the performances of two methods are compared and we analyze the advantages and disadvantages of each method in detail. To measure a loss rate, a low pass filter can be used instead of averaging, but we think that averaging is simple and enough to estimate.

B.2 Additional Overhead of VS-TCP

Since VS-TCP splits a connection at BS, the station has to manage two connections and buffer like the other split-connection schemes. Additionally, BS has to do more work, such as the segmentation of data and the segment-size control mechanism. When VS-TCP reduces the segment size, it should refine the segment whose size is larger. The refining process, which involves the memory copy from the segment to several small segments, has almost the same overhead of IP segmentation. To eliminate the overhead, a received segment is managed like a chain of several small memory blocks. Then the memory copy of the refining process can be replaced by the cut-off of the links between memory blocks. The overhead of the refining process can be removed by only managing some pointers.

VS-TCP adjusts the maximum segment size whenever it receives data or timeout occurs. The additional processing overhead with the segment-size control mechanism is negligible in comparison with the overall operation of TCP because varying the maximum segment size requires a few additions, subtractions and divisions. In normal TCP, the maximum segment size of distinct connections can be different. In other words, it already has data structure to support variable segment size. Hence, to implement the size variation mechanism, normal TCP needs to be changed little and the mechanism is simply added.

A possible overhead is that VS-TCP can invoke IP segmentation, whose cost is expensive. VS-TCP can sometimes make the segment the size of which exceeds the Path-MTU value and then IP segmentation may be exercised. To eliminate this problem, we disable IP segmentation and takes advantage of link-layer segmentation. Since the segmentation of link-layer occurs when the data is copied from the kernel memory to the memory of network interface, it is achieved with little overhead.

V. SIMULATION

A. Simulation Setup

We evaluated the performance of VS-TCP by means of simulation. Our simulation was based on the Network Simulator(*ns*) developed by the Lawrence Berkeley National Laboratory[11]. VS-TCP method 1 and 2 were compared with following schemes:

1. One connection : a normal TCP connection between a fixed host and a mobile host.
2. Split connection : It is I-TCP. It splits a TCP connection between a fixed host and a mobile host into two normal TCP connections. The one is between the fixed host and the base station and the other is between the base station and the mobile host.
3. Split connection with the modified TCP that has no congestion control : It is METP. It splits a TCP connection into two TCP connections. While a wired connection uses normal TCP, a wireless connection uses the modified TCP in which the congestion control is eliminated. The wireless TCP connection just retransmits lost packets. The pure performance enhancement due to varying the segment size can be extracted by comparing VS-TCP with

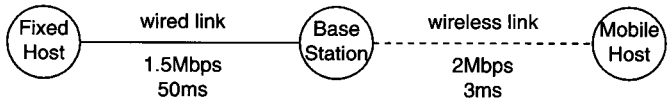


Fig. 5. The network topology in the simulation.

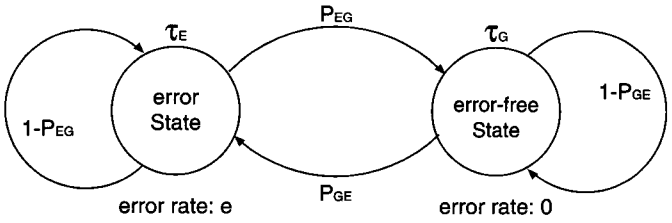


Fig. 6. Two states Markov model.

this scheme.

Each scheme is experimented 100 times during 30 minutes (in simulator time) and averaged.

A.1 Network Topology

The network topology used in our simulations is shown in Fig. 5.

There are three hosts, a fixed host in a wired network, a mobile host in a wireless network and a base station on the boundary. The fixed host and the mobile host communicate with each other via base station. The number below each link means the bandwidth and propagation delay of the link. These parameters are from the specification of wireless LAN and communication across the United States. The wireless link has its own bit-error pattern, that is explained in subsection V-A.2. We do not consider other traffic that pass through the links and hence all network resources are allocated to the TCP connections. An application in the fixed host establishes a TCP connection with the mobile host at the beginning of simulation and sends data continuously until the simulation finishes.

A.2 Error Model

To reflect the burstiness of the wireless error, we used two-state Markov model, shown in Fig. 6, for the error model of wireless link. This model consists of two states, error state and error-free state. The bit-error rate of error state is e and error-free state means no bit error. Error state is at least sustained during τ_E. After τ_E, The transition to error-free state can occur with probability of P_EG. If no state transition is decided, the state is again sustained during τ_E. In the consequence of above iteration, the error state is sustained during multiple τ_E. The average error state duration is calculated as follows:

$$\tau_E \cdot (1 + (1 - P_{EG}) + (1 - P_{EG})^2 + \cdots) = \frac{\tau_E}{P_{EG}}.$$

Accordingly, the burstiness of the error can be adjusted by the value of τ_E, P_{EG} and e. In the same manner, error-free state is at least maintained for τ_G and can be extended also.

Table 2 shows all parameter values used in simulations. τ_E, τ_G are obtained from the wave-LAN model proposed by Nguyen *et al.* [12]. Typically, τ_G, the duration of error-free state, was much longer than τ_E, the duration of error state. We changed the average packet loss rate of wireless link from 1% to 13% by adjusting P_{EG} from 0.1 to 0.4 and P_{GE} from 0.5 to 0.9; the packet loss rate is checked when the size of packet is 512 bytes.

Table 2. Parameter values for error model.

Parameter	Value
τ_E	0.0002 s
τ_G	0.0224 s
P_{EG}	$0.1 \sim 0.4$
P_{GE}	$0.5 \sim 0.9$
e	$0.003 \sim 0.01$

Table 3. Parameter values for VS-TCP.

Parameter	Value	Parameter	Value
s_i	512	Δs	128
s_{mx}	1024	s_{mn}	128
e_w	0.1	L_w	5

B. Simulation Result

By simulation, we compared the throughput of the proposed scheme and other schemes. Experiment results are shown in Fig. 7. In the figure, *one* means one connection over entire link, *two* does the split-connection using two normal TCP, and *nocon* does the split-connection with the modified TCP that has no congestion control. *VS-TCP 1* and *2* imply VS-TCP with method 1 and 2. The parameters used in *VS-TCP 1* and *2* were selected to perform the best and are shown in Table 3. The value of L_w, which determines time period to calculate average error rate, is chosen from the range between 1 and 100. Since short term average is more important than longer one in vulnerable wireless environment, values over 100 have little meaning. e_w is also selected to have maximum throughput by varying from 0 to 1.

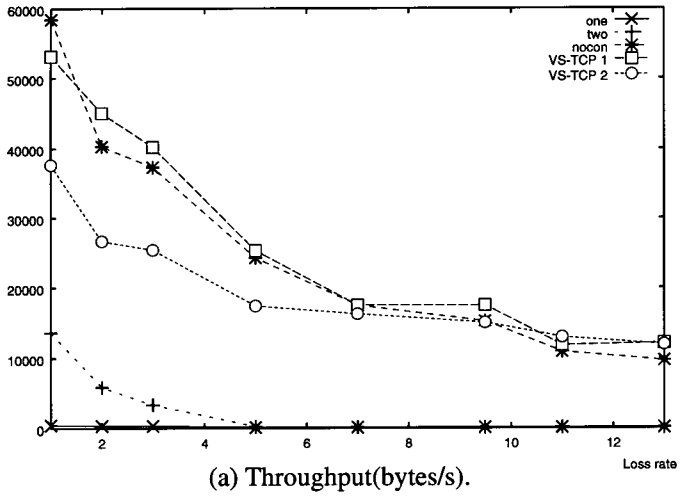
Fig. 7(a) shows the throughput of each scheme with varying the packet loss rate. The throughputs of all schemes decrease as the packet loss rate increases. To view the performance gap, throughputs normalized by that of *nocon* is plotted in Fig. 7(b). From that figure, the net performance gain of segment-size control mechanism can be obtained.

We could get following observations from the results:

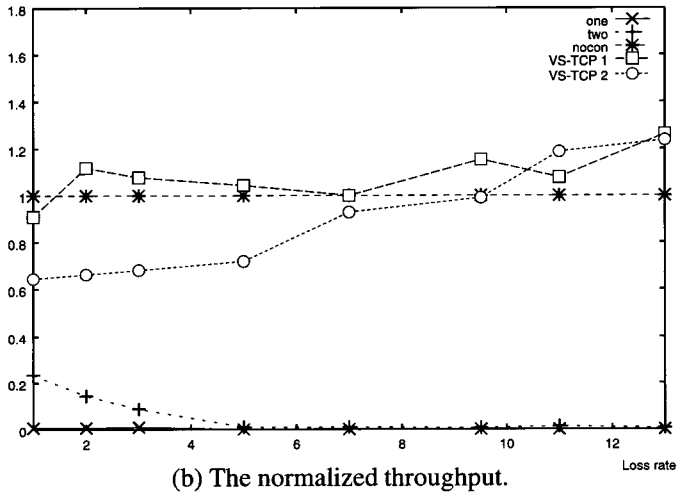
1. Schemes that do not have a congestion control mechanism, such as *nocon and VS-TCP 1 and 2*, outperform schemes that have.
   The reason is that the latter shrink window when a non-congestion loss occurs. The shrinkage of window leads to degradation of the throughputs. As the packet loss rate rises, they wrongly invoke the congestion control more frequently, and the performance consequently decreases.
2. When the packet loss rate is over 1%, the throughput of *VS-TCP 1* is better than that of *nocon*.
   When packets are frequently lost, VS-TCP reduces the segment size. As a result, the retransmission overhead is reduced and the probability of successful transmission increases. Reversely, when the packet loss rate is low, VS-TCP increases the segment size to reduce the header overhead. The performance gain of VS-TCP with respect to

Table 4. Optimal segment size for split-connection with wireless TCP that has no congestion control.

Packet error rate (%)	1	2	3	5	9.5	11
Optimal Segment size (bytes)	448	384	384	320	256	256

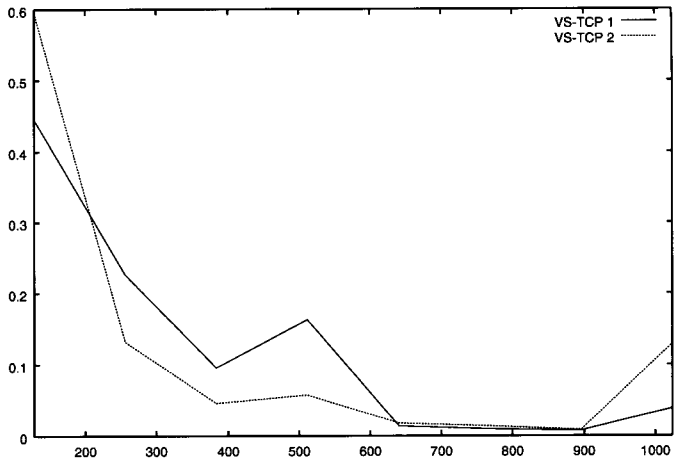


(a) Throughput(bytes/s).

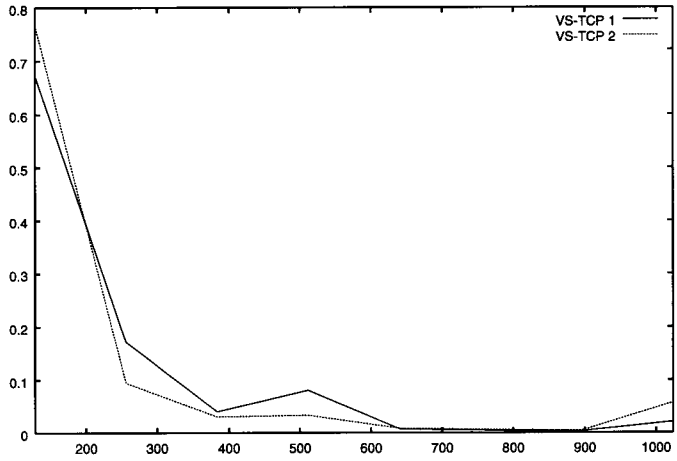


(b) The normalized throughput.

Fig. 7. The performance of each scheme with varying the packet loss rate.



(a) Packet loss rate 3%.



(b) Packet loss rate 11%.

Fig. 8. Segment size distribution of VS-TCP.

nocon comes from those actions.

The normalized throughput of VS-TCP may increase with the packet loss rate. Since the high packet loss rate means frequent long burst errors, the efficiency of the proposed segment-size control mechanism increases. Remember that the segment-size control mechanism conditionally diminishes the segment size by half. Inherently, there is an overhead of recovering the previous segment size. As the size of burst error becomes smaller, the effect of segment-size reduction decreases with the same recovering cost; then VS-TCP suffers more header overhead. In Table 4, the optimal segment size obtained is shown according to the packet loss rate. Therefore, VS-TCP works better when the packet loss rate is higher.

In typical wireless networks, such as wave-LAN and CDPD, the average packet loss rates are 3% and 5% respectively. [4], [12] From the results, we confirm that VS-TCP probably outperforms than *nocon* in a real case.

3. *VS-TCP 1* has better performance than *VS-TCP 2*, and the throughput gap between them decreases as the packet loss rate increases.

The reason is that, in our simulation environment, the variation of loss rate is not small so that large value of L_w cannot be used. With large L_w, e.g., 100, *VS-TCP 2* may fail to follow up current loss rate in time. By experiments, we found that the optimal value of L_w is 5. However, with small L_w, calculated loss rate is too coarse. For example, only one loss results in the calculated loss rate of 20%. If the real loss rate is small, the difference between the calculated loss rate and the real loss rate is large enough to degrade the performance of *VS-TCP 2* even worse than *VS-TCP 1*. As the loss rate becomes larger, however, the difference is reduced and the *VS-TCP 2*'s advantage, which is to check the more accurate error rate, takes effects.

VI. CONCLUSIONS AND FUTURE WORKS

We suggested a novel transport protocol named VS-TCP in order to enhance TCP performance over wireless networks, where non-congestion packet losses occur frequently. The unique feature that makes VS-TCP distinct from other wireless transport protocols is the non-congestion loss control mechanism. The others, such as I-TCP, METP, and WTCP, focused on distinguishing the type of loss so that they apply the congestion control mechanism only to congestion losses. Our scheme also discriminates the loss type by splitting a connection. In addition, VS-TCP applies the proposed non-congestion loss control function to non-congestion losses. The non-congestion control mechanism adjusts the segment size according to the current non-congestion packet loss rate: If packets are frequently lost, it decreases the segment size so as to reduce the retransmission overhead and the probability of packet corruption. When the packet loss rarely occurs, the segment size is enlarged to reduce the header overhead. To study the proposed protocol, we compared it to others by simulations. Our results showed that VS-TCP outperforms others about 10% on the average, up to 30%. VS-TCP 1 has the best performance in typical wireless environments. VS-TCP 2 works well when the packet loss rate is over 10%.

Since the loss-type discrimination of VS-TCP is based on a split-connection mechanism, it has same problems, such as the overheads of BS and hand-off. However, the non-congestion control mechanism of VS-TCP can be used with any scheme that can distinguish non-congestion losses from others. In our future work, we will merge the non-congestion control mechanism with end-to-end schemes and, at last, design a novel transport protocol that has its own mechanisms handling the losses of both types.

REFERENCES

[1] J. Postel, *Transmission Control Protocol*, RFC 793, Sept. 1981.

[2] A. Bakre and B. Badrinath, "I-TCP: Indirect TCP for mobile hosts," in *Proc Int. Conf. Distributed Computing Systems*, Vancouver, Canada, May 1995.

[3] K. Y. Wang and S. K. Tripathi, "Mobile-end transport protocol: An alternative to TCP/IP over wireless links," in *Proc. IEEE INFOCOM '98*, 1998, pp. 1046–1053.

[4] K. Ratnam and I. Matta, "WTCP: An Efficient mechanism for improving TCP performance over wireless links," *Computers and Commun.*, pp. 74–78, 1998.

[5] T. Goff *et al.*, "Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments," in *Proc. IEEE INFOCOM 2000*, 2000, pp. 1537–1545.

[6] R. Ludwig, A. Konrad, and D. Joseph, "Optimizing the end-to-end performance of reliable flows over wireless links," in *Proc. ACM Mobicom '99*, 1999, pp. 113–119.

[7] W. R. Stevens, *TCP/IP Illustrated, Volume 1 (The Protocols)*, Addison Wesley, Nov. 1994.

[8] C. Kent and J. Mogul, "Fragmentation considered harmful," *ACM SIGCOMM Computer Commun. Review*, vol. 17, Issue. 5, Aug. 1987.

[9] S. Keshav, *An Engineering Approach to Computer Networking*, Addison Wesley, Sept. 1997.

[10] P. Bhagwat *et al.*, "Enhancing throughput over wireless LANs using channel state dependent packet scheduling," in *Proc. IEEE INFOCOM 1996*, 1996.

[11] Network Research Group, Lawrence Berkeley National Laboratory, *ns-LBNL Network Simulator*, Available at http://www.isi.edu/nsnam/ns/.

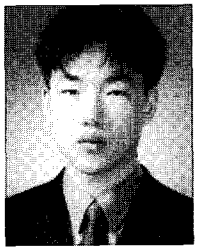
[12] G. T. Nguyen and B. Noble, "A trace-based approach for modeling wireless channel behavior," in *Proc. Winter Simulation Conf.*, Dec. 1996.

[13] B. S. Bakshi *et al.*, "Improving performance of TCP over wireless networks," *IEEE Distributed Computing Systems*, 1997.

[14] L. S. Brakmo, S. O' Malley, and L. L. Peterson, "TCP vegas: New techniques for congestion detection and avoidance," in *Proc. ACM SIGCOMM*, pp. 24–35, Oct. 1994.

[15] M. Mathis and J. Mahdavi, "Forward acknowledgment: Refining TCP congestion control," in *Proc. ACM SIGCOMM*, Aug. 1996.

[16] M. Mathis *et al.*, *TCP Selective Acknowledgment Options*, RFC 2018 edition, 1996.



Keuntae Park received his B.S. and M.S. degrees in electrical and electronic engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 1999 and 2001, respectively. Currently he is a Ph.D. candidate of the department of electrical engineering and computer science in KAIST, since 2001. His research interests include protocol design, analysis and implementation for wireless network.

Sangho Park received his B.S. and M.S. degrees in electrical and electronic engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 1997 and 1999, respectively. Currently he is a Ph.D. candidate of the department of electrical engineering and computer science in KAIST, since 1999. His research interests include protocol design, analysis and implementation for wireless network.

Daeyeon Park received his B.S. and M.S. degrees in computer science from University of Oregon, USA, in 1989 and 1991, respectively and Ph.D. degree in computer science from University of Southern California, USA, 1996. He worked at University of Korea Foreign Language from 1996 and 1997. He joined the Department of Electrical Engineering at KAIST in 1998, where he is currently an Assistant Professor. His major interests include operation system, distributed system, parallel processing and mobile ad hoc network. He is a member of KIEE, KISS and IEEE.