Wireless Networks 6 (2000) 191–199

A new ATM adaptation layer for TCP/IP over wireless ATM networks

Ian F. Akyildiz [a] and Inwhee Joe [b]

[a] *Broadband and Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA*

[b] *Network Research Group, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA*

This paper describes the design and performance of a new ATM adaptation layer protocol (AAL-T) for improving TCP performance over wireless ATM networks. The wireless links are characterized by higher error rates and burstier error patterns in comparison with the fiber links for which ATM was introduced in the beginning. Since the low performance of TCP over wireless ATM networks is mainly due to the fact that TCP always responds to all packet losses by congestion control, the key idea in the design is to push the error control portion of TCP to the AAL layer so that TCP is only responsible for congestion control. The AAL-T is based on a novel and reliable ARQ mechanism to support quality-critical TCP traffic over wireless ATM networks. The proposed AAL protocol has been validated using the OPNET tool with the simulated wireless ATM network. The simulation results show that the AAL-T provides higher throughput for TCP over wireless ATM networks compared to the existing approach of TCP with AAL 5.

1. Introduction

Current TCP (Transmission Control Protocol) implementations contain several new algorithms to improve TCP performance on packet losses. For example, TCP-Tahoe uses the *fast retransmit* algorithm [10] and TCP-Reno adds the *fast recovery* algorithm [11], while TCP-Vegas attempts to provide earlier detection of packet losses and accurate round-trip delay estimation [5]. However, all of these implementations are optimized for the case when a single packet is dropped from a single window. If multiple packets are dropped from a single window, their performance suffers severely, because the sender is forced to recover by means of a retransmission timeout instead of fast recovery. For TCP over ATM (Asynchronous Transfer Mode) networks, cell losses in ATM switches may span TCP packet boundaries, resulting in the loss of two TCP packets. Moreover, when ATM networks include wireless links, multiple packet losses can occur due to higher and burstier error patterns of wireless links.

Recently, SACK (Selective Acknowledgment) and New-Reno have been proposed to improve TCP performance when multiple packet losses occur within a single window [8,13]. However, TCP does not have to shrink its congestion window at all in response to the packet losses due to link errors or handoffs, because such losses have nothing to do with network congestion. Obviously, this unnecessary window shrinking causes significant performance degradation. The fundamental solution to this problem is to distinguish between link errors and network congestion. Therefore, our approach is to push the error control portion of TCP down to the ATM adaptation layer (AAL) so that TCP is only responsible for congestion control. As a result, TCP does not invoke congestion control mechanism under any circumstances except for actual network congestions.

Currently, AAL 5 is being used to transport data traffic (e.g., frame relay and IP packets) in ATM networks, since it is much simpler and efficient compared to AAL 3/4 [9]. Furthermore, AAL 5 has the following advantages: the wide acceptance of AAL 5 from both computer and telecommunication industries, no requirement for extra hardware, (possibly) easy extension to support variable bit rate (VBR) video and audio transport, and effective error handling. However, the use of AAL 5 cannot be considered as the optimal solution for wireless ATM in terms of error characteristics, because there can be still severe discard at AAL level due to residual link errors or handoffs.

In this paper, we propose a new AAL protocol, AAL-T, to improve TCP performance over wireless ATM networks. To support quality-critical TCP traffic over wireless ATM networks, the AAL-T protocol is based on a new and very efficient ARQ (Automatic Repeat Request) scheme, which does not have timers and retransmits when the packet is indeed lost or in error [1]. In the next section, we discuss TCP problems over wireless ATM networks, followed by a detailed design of AAL-T. In section 4 we analyze throughput efficiency for TCP versus AAL-T. In section 5 we present our simulation models and performance evaluation results from OPNET simulation. Finally, we conclude the paper by highlighting our contribution.

2. Issues of TCP over wireless ATM

TCP is a widely used transport protocol in wireline networks for reliable communications. However, the use of TCP over wireless ATM networks introduces several problems, which have been studied in detail recently [14,19]. TCP uses the Go-Back-N (GBN) ARQ scheme with positive acknowledgment (ACK). The protocol also uses

a 16 bit checksum for error detection, and sequence number for detection of lost, duplicate, and out-of-sequence packets. As a result of this scheme, when there is no ACK from the receiver, TCP cannot distinguish link errors from network congestion. In response to all packet losses, TCP always invokes congestion control mechanism to shrink the window size and to increase the retransmission timeout (RTO). This leads to severely reduced throughput and often a complete connection timeout condition.

For congestion control, the basic mechanisms in TCP are as follows. First, TCP detects a packet loss through a timeout event. TCP shrinks its window down to one upon packet loss detection. Subsequently, the window grows rapidly, by one packet for every successfully acknowledged packet, until it reaches half of the window size at the last packet loss. This stage of window growth is called *slow start*, since it is slow compared to not having decreased the window at all after a loss. After slow start, the algorithm switches to *congestion avoidance*, in which the window grows slowly in order to probe for extra bandwidth, by incrementing the window size by one for every window's worth of acknowledged packets. This growth continues until the maximum window size is reached, or until another packet loss is detected.

Since TCP uses the GBN-ARQ scheme, the transmitter should resend N previous packets instead of sending new packets, whenever there is no ACK from the receiver. It may result in significant performance degradation, especially when the scheme is used with high error-rate links such as wireless links. Another factor that makes TCP inadequate is the window size. It is limited by the 16-bit window field in the TCP header, because TCP implements a credit mechanism using the window field to advertise to the transmitter how much buffer space (bytes) is available at the receiver. The default value for the TCP window size is only 16 Kbytes which can severely restrict the throughput efficiency for satellite networks in particular.

TCP cannot effectively cope with a network where the network capacity changes rapidly. Link errors result in AAL discard, and thus "holes" in the TCP segment stream. These holes have the same effect as a dynamic change of network capacity to TCP. Furthermore, TCP performs poorly in response to unpredictable error bursts, which are typical characteristics of wireless links. Burst errors often halt the TCP connection completely, resulting in the TCP applications being locked out for many minutes until new connection is re-established.

Finally, the poor performance of TCP over ATM has been observed under congestion conditions [15]. There are two possible service classes for data traffic like TCP traffic over ATM networks: ABR (Available Bit Rate) and UBR (Unspecified Bit Rate). In the ABR service, two separate congestion control methods are used, one at the transport layer (TCP) and the other at the ATM layer (ABR). The effectiveness can be reduced by the mismatch between these two methods, i.e., the source rate at the ATM layer does not follow exactly the TCP window size [12]. When the UBR service is used, the TCP protocol is responsible for congestion control. In this case, however, channel utilization may drop if concurrent TCP connections are sharing a single link in the ATM network, because the window sizes for TCP connections are shrunk at the same time during the congestion period.

For quality-critical traffic, the use of an appropriate AAL is required when TCP is used over wireless ATM networks. This AAL will take care of error control by retransmission, while TCP is responsible for congestion control by using the UBR service at the ATM layer. As a result, TCP does not invoke congestion control mechanism under any circumstances except for real network congestion. Furthermore, since TCP does not need retransmission, this approach avoids redundant retransmissions and reduces the TCP complexity by eliminating the retransmission timer. In contrast to the GBN-ARQ scheme of TCP, the new AAL protocol will be based on the Selective-Repeat (SR) ARQ scheme to obtain better throughput efficiency by retransmitting only packets in error. For a large window size, the new AAL protocol will feature a large sequence number space.

3. The AAL-T design

As discussed in section 2, TCP introduces significant performance degradation over wireless ATM networks with AAL 5. To resolve this problem, we propose the AAL-T to support quality-critical traffic (e.g., data, imagery) over high bit-error-rate (BER) wireless links by using an ARQ-based retransmission scheme. As shown in figure 1, AAL-T consists of two sublayers: Reliable Convergence Sublayer (R-CS) and Segmentation and Reassembly Sublayer (SAR). Unlike the existing AAL protocols, R-CS is further subdivided into two sublayers: Common Part Convergence Sublayer (CPCS) and Common Part ARQ (CP-ARQ) sublayers. The CP-ARQ sublayer is a new sublayer introduced to take care of end-to-end error control, so that the CPCS sublayer can pass error-free data to the TCP layer.

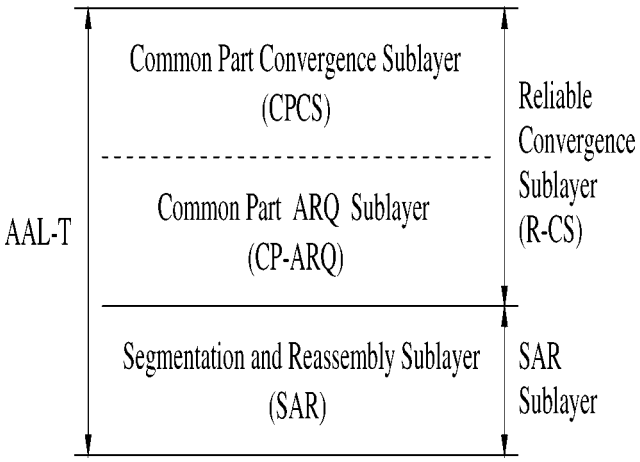

Figure 1. AAL-T protocol structure.

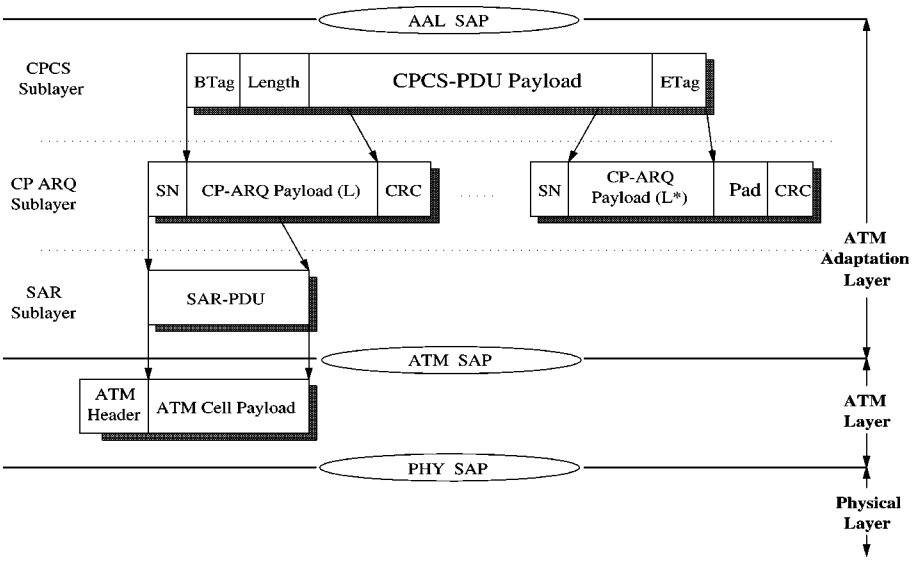

Figure 2. Segment formats of the AAL-T.

Since the retransmission timer is not used in TCP over AAL-T, we need another mechanism to invoke the congestion control method during network congestion. The ATM network management detects congestion conditions by monitoring RM (Resource Management) cells, or EFCI (Explicit Forward Congestion Indication) bits in the ATM cell header in case RM cells are not available. Then, an interface routine is used to give a congestion notification to the ICMP (Internet Control Message Protocol) layer at the destination so that ICMP can send an error message ("Source Quench") to the source TCP. In response to the ICMP message, the source TCP invokes congestion control specified in RFC 1122. Our approach does not violate the TCP/IP standard, because the interface routine is a local function within a host.

Another problem with the UBR service is that channel utilization drops drastically because the window sizes for TCP connections are shrunk at the same time during the congestion period. One solution to this problem is window scattering proposed in [15]. Scattering the windows for TCP connections can be easily implemented in AAL-T by sending a randomly delayed notification to each TCP connection. This approach results in graceful degradation in terms of aggregate bandwidth to the ATM network.

3.1. The common part convergence sublayer (CPCS)

The CPCS-PDU (Protocol Data Unit) format is shown in figure 2, which includes a header of three octets. The header consists of two fields to indicate the beginning of the CPCS-PDU by *BTag* (1 byte) and the number of octets of the CPCS-PDU by *Length* (2 bytes). The length field is used as the final checking to ensure that the CPCS sublayer passes error-free and orderly data to the TCP layer. Section 3.4 will provide details for robustness of AAL-T. The CPCS-PDU also includes a trailer (the 1-byte *ETag* field) to indicate the end of the packet.

When the CPCS sublayer receives the last CP-ARQ PDU, it tells the CP-ARQ sublayer to send the control packet immediately without waiting for next control packet transmission. This allows a quick response for small-amount data transmission.

3.2. The common part ARQ (CP-ARQ) sublayer

The CP-ARQ sublayer provides a reliable end-to-end transmission capability for wireless ATM networks. The ARQ scheme deals with all types of errors by using retransmission. Two types of errors are considered here: residual errors that are not captured by error control methods at the lower layers, and cell losses due to handoff or network congestion resulting from the statistical multiplexing of ATM.

The CP-ARQ PDU format is shown in figure 2. The *Sequence Number* (SN) field is used to deliver the CP-ARQ PDUs in order. If a CP-ARQ PDU is lost or in error, the CP-ARQ layer will attempt to retransmit the PDU with the same SN in the buffer. The size of the SN field is 2 octets, which is large enough to support the window size even for satellite networks. The Cyclic Redundancy Check (CRC-16) is used for error detection.

Except for the last CP-ARQ PDU, the payload length of the CP-ARQ PDU is denoted by L, where L is a step function of the end-to-end path conditions. The end-to-end path BER can be calculated at the CP-ARQ sublayer from the error statistics of the most recently received packets by comparing error packets and retransmitted correct packets. The end-to-end path condition is dominated by the low quality wireless links. For the last CP-ARQ packet, the payload length is denoted by L^*, which covers the rest of the CPCS-PDU. The *Pad* field ensures that the last CP-ARQ PDU is also a multiple of SAR-PDU (48 bytes). The *Pad* field is 0–43 octets, which can be calculated from the length of the header and trailer of the CP-ARQ PDU.

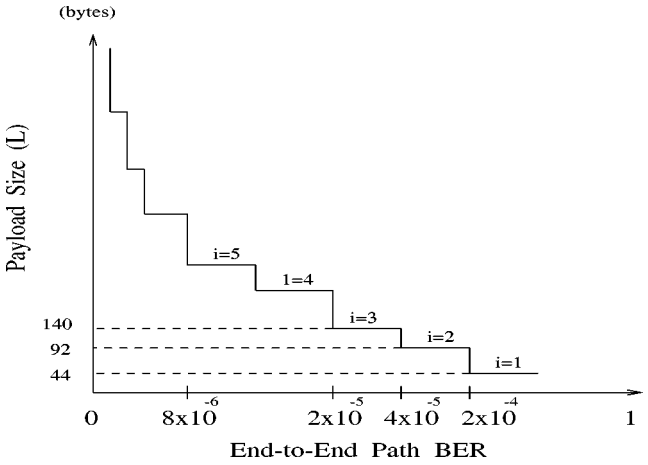

Figure 3. Look-up diagram for payload lengths versus end-to-end path BER.

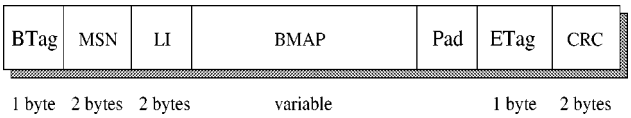

Figure 4. Control packet format.

The payload length of the CP-ARQ PDU, L, is dynamically updated according to a look-up table which computes the payload length as a function of the end-to-end path BER. Figure 3 shows the logical look-up table where $L = 48i-4$ for $i = 1, 2, 3, \ldots$. Initially, $i = 1$ will be used for the worst case, and the transmitter updates the payload length L every time it receives the control packet from the receiver. Based on the new payload length, the receiver updates the generation rate of control packets. The transmitter does not take any action for the new control packet generation rate. This well-defined procedure easily solves the efficiency problem even when the quality of an end-to-end path is poor.

The Survivable ARQ (S-ARQ) is used at the CP-ARQ sublayer to offer higher efficiency than the other ARQ-based schemes by not dealing with timers. The S-ARQ will be briefly explained in section 3.2.1.

3.2.1. The survivable ARQ (S-ARQ)

Unlike classical ARQ schemes, the Survivable ARQ scheme does not have any retransmission timer to prevent redundant retransmissions and long recovery periods from a timeout mechanism. The key ideas of S-ARQ are "variable packet size" and "periodic status message". The packet size varies with the end-to-end path conditions so as to maximize the throughput efficiency. The receiver sends its status to the transmitter on a periodic basis by control packets, thereby eliminating the timeout mechanism.

The control packet structure is shown in figure 4, which consists of seven fields: *BTag* and *ETag* have just the same bit patterns as in the CPCS-PDU, *MSN* is the maximum sequence number of the CP-ARQ PDU below which every PDU has been received correctly, *BMAP* is a bit map indicating outstanding CP-ARQ PDUs between *MSN* and the last received PDU, *LI* is the length indicator for *BMAP* in bits, *Pad* makes the control packet a multiple of SAR-PDU (48 bytes), and CRC-16 is used for error detection in the control packet.

Another feature of S-ARQ is to provide negative and positive feedback by utilizing *BMAP*. Upon receipt of a control packet, the transmitter marks all CP-ARQ PDUs preceding *MSN* as successfully transmitted. The PDUs that are set to one in the *BMAP* field are also marked as successfully transmitted through the positive acknowledgment of S-ARQ. For the PDUs that are set to zero in *BMAP* and have been transmitted, the m value is decreased by one, where m indicates the frequency of the control packet per round trip delay and its initial value is determined with the simulation results by considering the tradeoff between response time and bandwidth [1]. If this m value goes to zero, then the round trip delay has elapsed, so the PDU is retransmitted according to the negative acknowledgment. An estimate of the round trip delay *srt* is given by [16]

$$srt_i = \alpha \cdot srt_{i-1} + (1-\alpha) \cdot rt_i, \tag{1}$$

where α is a smoothing constant ranging from 0 to 1 (e.g., usually 0.9 for TCP) and rt_i is the round trip delay measured for the ith CP-ARQ packet.

In addition, S-ARQ provides multiple and selective acknowledgments by turning on bits in *BMAP* for packets received correctly at the receiver. Since TCP only acknowledges packets in order, TCP may experience poor performance when multiple packets are lost from one window of data. In fact, TCP takes one round trip time to find out about each lost packet, while S-ARQ takes care of multiple packet losses in one round trip time by using *BMAP*.

The performance of ARQ schemes is very sensitive to the packet size. It is apparent that if the packet size is too small, the protocol is operating inefficiently because of high overhead per packet. On the contrary, if the packet size is too large, the packet is more likely to be received in error, resulting in more retransmissions and poor throughput. Therefore, there exists an optimal packet size that maximizes the throughput efficiency of the protocol. The optimal choice is found to depend on the error characteristics of the channel and the number of overhead bits used for control. Since the SR-ARQ scheme only retransmits packets in error, the optimal information payload length is given in [18]

$$L_{\text{opt}} = \frac{-h\ln(1-P_{\text{b}}) - \sqrt{-4h\ln(1-P_{\text{b}}) + h^2\ln(1-P_{\text{b}})^2}}{2\ln(1-P_{\text{b}})}, \tag{2}$$

where P_{b} is the end-to-end path BER and h is the number of overhead bits per CP-ARQ packet. In S-ARQ, the payload length is dynamically updated according to the end-to-end path conditions in order to maximize the throughput efficiency. Since the overhead size h is fixed in the CP-ARQ

Table 1
CP-ARQ payload table.

End-to-end BER	Payload (L)	Cells
$2 \cdot 10^{-4}$ or higher	44 bytes	1
$4 \cdot 10^{-5} - 1 \cdot 10^{-4}$	92 bytes	2
$2 \cdot 10^{-5} - 3 \cdot 10^{-5}$	140 bytes	3
$8 \cdot 10^{-6} - 1 \cdot 10^{-5}$	236 bytes	5
$6 \cdot 10^{-6} - 7 \cdot 10^{-6}$	284 bytes	6
$4 \cdot 10^{-6} - 5 \cdot 10^{-6}$	332 bytes	7
$3 \cdot 10^{-6}$	428 bytes	9
$2 \cdot 10^{-6}$	476 bytes	10
$1 \cdot 10^{-6}$	716 bytes	15

packet, the optimal payload length L_{opt} depends entirely on the end-to-end path BER at that time. Given the 32-bit overhead size used in the CP-ARQ packet, the actual payload length of the CP-ARQ packet, L, is chosen as the closest value to L_{opt} that makes the CP-ARQ PDU a multiple of SAR-PDU (48 bytes), as shown in table 1.

3.3. The segmentation and reassembly (SAR) sublayer

The SAR sublayer function is adopted from AAL 5. The SAR-PDU consists of 48 octets of payload, carrying a portion of the CP-ARQ PDU. The ATM-user-to-ATM-user (AAU) bit, the last bit in the payload-type field (3 bits) of the ATM cell header, is used to indicate which portion of the CP-ARQ PDU is contained in a SAR-PDU. That is, the AAU bit is set to one for the last cell of a CS-ARQ PDU, and set to zero for all other cells. The process is the same as in AAL 5, except that the AAU bit is used to delineate the CP-ARQ PDU in AAL-T instead of the CPCS-PDU.

3.4. Robustness of AAL-T

In this section, we investigate some typical scenarios to check if AAL-T layer is robust.

Scenario 1. Suppose that the first cell or any middle cell (AAU = 0) of the CP-ARQ PDU is lost or damaged: CRC-16 detects the error after receiving the last cell of the CP-ARQ PDU, and the receiver will ask for retransmission of this CP-ARQ PDU.

Scenario 2. Suppose that the last cell (AAU = 1) of the CP-ARQ PDU is damaged: CRC-16 detects the error after receiving the last cell of the CP-ARQ PDU, and the receiver will ask for retransmission of this CP-ARQ PDU.

Scenario 3. Suppose that the last cell (AAU = 1) of the CP-ARQ PDU is lost. The CP-ARQ sublayer cannot notice that a cell is missing until the receiver gets the last cell of the next CP-ARQ PDU. After that, CRC-16 detects the error and the receiver will ask for retransmission of the first CP-ARQ PDU. For the second CP-ARQ PDU, it will ask for retransmission by the missing sequence number after receiving another subsequent CP-ARQ PDU.

Scenario 4. Suppose that the first and last cells of the CP-ARQ PDU are both lost. Like scenario 3, the CP-ARQ sublayer cannot notice any problem until the receiver gets the last cell of the next CP-ARQ PDU. After that, CRC-16 detects the error, but the receiver will not ask for retransmission because of the wrong sequence number. Only after receiving another subsequent CP-ARQ PDU, the receiver can ask for retransmission of both of the previous two CP-ARQ PDUs by the missing sequence numbers.

Scenario 5. Suppose that all cells of a CP-ARQ PDU are lost. The sequence number in the CP-ARQ PDU header catches the missing CP-ARQ PDU, and the receiver will ask for retransmission of this CP-ARQ PDU.

Scenario 6. Suppose that any octet in the CPCS-PDU is matched with the *BTag* or the *ETag* field by chance. This problem can be solved by the length field in the CPCS-PDU header.

Scenario 7. CRC Failure Case. First, consider the probability of CRC failure. For CRC-16, the undetected error probability of burst errors of length larger than 17 is 0.0015%, which is very low. However, it is true that the CRC could fail to detect errors although the probability is very small. One solution to that is to add a length field to the CP-ARQ PDU. Since the length field needs about 2 bytes, the overhead size becomes 6 bytes for each CP-ARQ PDU, which is too high, when considering the entire overhead including a 20 byte TCP header, a 20 byte IP header, and a variable padding length (0–43 bytes).

The length field in the CPCS-PDU header is used as a final gate for error detection. For example, the total overhead size in AAL-T is 8 bytes (i.e., 4 byte overhead from the CPCS sublayer plus 4 byte overhead from the CP-ARQ sublayer) for the best case, which is the same as in the AAL 5. When the CPCS sublayer detects an error by the length field, it will send a retransmission request for the entire CPCS-PDU to the transmitter. Of course, the transmitter needs to hold CPCS-PDUs in the buffer until the acknowledgment for the CPCS-PDU arrives from the receiver. Since the buffer size is a multiple (e.g., 4) of the window size in the S-ARQ scheme, there is no buffer overflow in practice. In order to acknowledge each CPCS-PDU, the tag fields (*BTag* and *ETag*) are used to identify CPCS-PDUs. For each CPCS-PDU, a unique bit sequence is assigned.

4. Performance analysis

The *throughput* of an ARQ protocol is defined as the ratio of the number of successful transmissions to the total number of transmissions including retransmissions in terms of data packets. The *transmission efficiency* is defined as the fraction of the payload size to the total packet size. The *throughput efficiency* is the *throughput* multiplied by the *transmission efficiency*. We analyze the throughput efficiency of TCP/IP over AAL 5 as follows.

For the GBN-ARQ protocol such as used in TCP, the *expected number of transmission attempts* per packet is given by

$$E[N] = \sum_{i=1}^{\infty} \big(N_{\mathrm{w}}(i-1)+1\big)(1-P_{\mathrm{s}})^{i-1}P_{\mathrm{s}} = \frac{P_{\mathrm{s}} + (1-P_{\mathrm{s}})N_{\mathrm{w}}}{P_{\mathrm{s}}}, \tag{3}$$

where N_{w} is the window size and P_{s} is the probability of successful transmission of a data packet. From equation (3), the *throughput efficiency* of TCP, η_{TCP}, is computed by

$$\eta_{\mathrm{TCP}} = \left(\frac{1}{E[N]}\right)\cdot\left(\frac{l}{l+h}\right) = \frac{P_{\mathrm{s}}}{P_{\mathrm{s}} + (1-P_{\mathrm{s}})N_{\mathrm{w}}}\cdot\left(\frac{l}{l+h}\right), \tag{4}$$

where l is the payload size and h is the overhead size per packet. As shown in equation (4), the throughput efficiency consists of two terms: the first term represents the throughput of the protocol, while the second term represents the transmission efficiency.

For the SR-ARQ protocol such as used in AAL-T, the throughput efficiency is a special case of equation (4) with the window size of one ($N_{\mathrm{w}} = 1$), because only a packet in error is sent for each retransmission:

$$\eta_{\mathrm{AALT}} = P_{\mathrm{s}}\cdot\left(\frac{l}{l+h}\right). \tag{5}$$

If one packet consists of m ATM cells, P_{s} can be expressed as

$$P_{\mathrm{s}} = (P_{\mathrm{c}})^m, \tag{6}$$

where P_{c} is the probability that a cell is received correctly. In order to derive P_{c} in terms of BER, consider the cell loss probability (CLR) when the HEC (Header Error Control) function operates in the correction/detection mode. Note that cell loss results from the ATM cell header in error rather than the ATM cell payload. With the assumption of ideal interleaving, bit errors would occur independently and hence CLR is given by

$$\mathrm{CLR} = P_{\mathrm{cm}}P_{\mathrm{m}} + P_{\mathrm{dm}}(P_1 + P_{\mathrm{m}}) = 1 - P_0 - P_0P_1, \tag{7}$$

where

- $P_{\mathrm{cm}} = P_0$ is the probability in the correction mode,
- $P_{\mathrm{dm}} = 1 - P_{\mathrm{cm}}$ is the probability in the detection mode,
- $P_0 = (1-P_{\mathrm{b}})^{40}$ is the probability of no error in an ATM cell header,
- $P_1 = 40(1-P_{\mathrm{b}})^{39}P_{\mathrm{b}}$ is the probability of a single-bit error in an ATM cell header,
- $P_{\mathrm{m}} = 1 - P_0 - P_1$ is the probability of multiple-bit errors in an ATM cell header,
- P_{b} is the probability of bit error in terms of end-to-end path.

P_{c} can be calculated as the probability that both the cell header and the cell payload are correct. Since these two events are independent, P_{c} is given as the product of two probabilities:

$$\begin{aligned} P_{\mathrm{c}} &= (1-\mathrm{CLR})(1-P_{\mathrm{b}})^{384} \\ &= (P_0 + P_0P_1)(1-P_{\mathrm{b}})^{384} \\ &= (1-P_{\mathrm{b}})^{424}\big(1+40P_{\mathrm{b}}(1-P_{\mathrm{b}})^{39}\big), \end{aligned} \tag{8}$$

where 384 is the ATM cell payload size in bits. In addition, P_{c} can be simply expressed in terms of CER (Cell Error Rate):

$$P_{\mathrm{c}} = 1 - \mathrm{CER}, \tag{9}$$

where CER is defined as the percentage of errored or lost cells to transmitted cells.

TCP/IP over AAL 5 has at least a total overhead of 48 bytes: 20 bytes of TCP header, 20 bytes of IP header, and 8 bytes of AAL 5 trailer. Also, there is a variable length of padding (0–47 bytes). Like AAL 5, AAL-T has minimum 48 bytes of overhead. The difference is that the padding length in AAL-T is (0–43) bytes. When the payload size of the TCP packet is 1200 bytes, which is typical, there is no need for padding because it is exactly a multiple of 48 bytes. In this case, the number of cells per packet is 26. Given parameters above, the throughput efficiency of AAL-T is compared with the TCP case for various window sizes as shown in figures 5(a) and (b) as a function of BER and CER, respectively. The results show that the AAL-T protocol always outperforms the TCP protocol for all BER or CER ranges. For example, the throughput efficiency of AAL-T is 0.94 at a CER of 10^{-3}, which corresponds to about 20% improvement compared to the TCP case with a window size of 10 packets at the same CER. In summary, since TCP exhibits poor performance for high error rates in a wireless ATM path, AAL-T is justified as a viable solution for wireless ATM networks.

5. Simulation

The objective of our simulation is to evaluate the AAL-T protocol to support the reliable transport for quality-critical traffic using TCP/IP over wireless ATM networks. We compare the performance of AAL-T with the TCP/IP on top of AAL 5 and demonstrate the improvements achieved by AAL-T.

5.1. Simulation model

We develop our network simulation models using MIL3's OPNET simulation package. As shown in figure 6, the network model consists of six ATM switches. Each ATM switch supports 8 cross connections and the link speed is OC-3 (155 Mbps). Since the TCP/IP traffic is connectionless data, the service class is classified as class D where no time relation exists between the source and the destination, and the bit rate is variable. In each ATM switch,

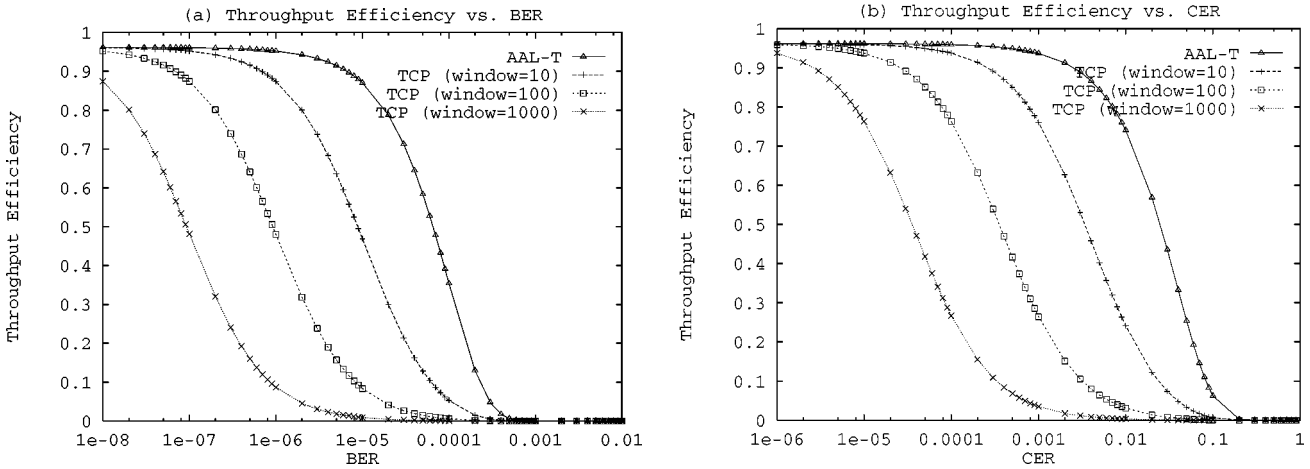

Figure 5. Comparison of AAL-T and TCP in terms of throughput efficiency.

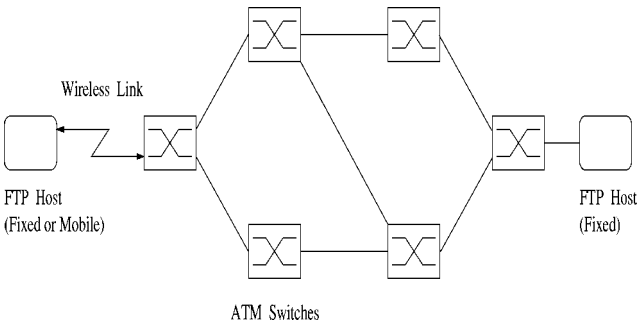

Figure 6. Simulation model.

the buffer size of 1000 cells is reserved for class D traffic. Switching delays in the ATM switch and propagation delays are assumed to be negligible. The wireless channel is modeled as a Rayleigh fading channel combined with AWGN (Additive White Gaussian Noise) generator.

In order to evaluate the performance of TCP/AAL 5 versus AAL-T, we conduct file transfer tests using FTP (File Transfer Protocol), which is a typical TCP application. The TCP version in our simulation is based on the TCP-Tahoe release with slow start and congestion avoidance. The receiver buffer size in TCP is 64 Kbytes and the maximum packet size is 9180 bytes of the default value for IP over ATM networks. For AAL 5, the initial retransmission timer value is set to 1 s. When the retransmission timer expires, TCP always triggers slow start and retransmission at the same time as in the standard approach of AAL 5.

We develop two simulation models as shown in figure 7:

- *Configuration 1*: TCP/IP and AAL 5 (as the standard in OPNET).
- *Configuration 2*: we keep the congestion control in TCP/IP and shift the error control function of TCP to AAL-T.

Since AAL-T takes care of error control in configuration 2, TCP does not have to request retransmissions. We can easily eliminate the retransmission part from the TCP protocol by setting the timer value to infinity. As a result,

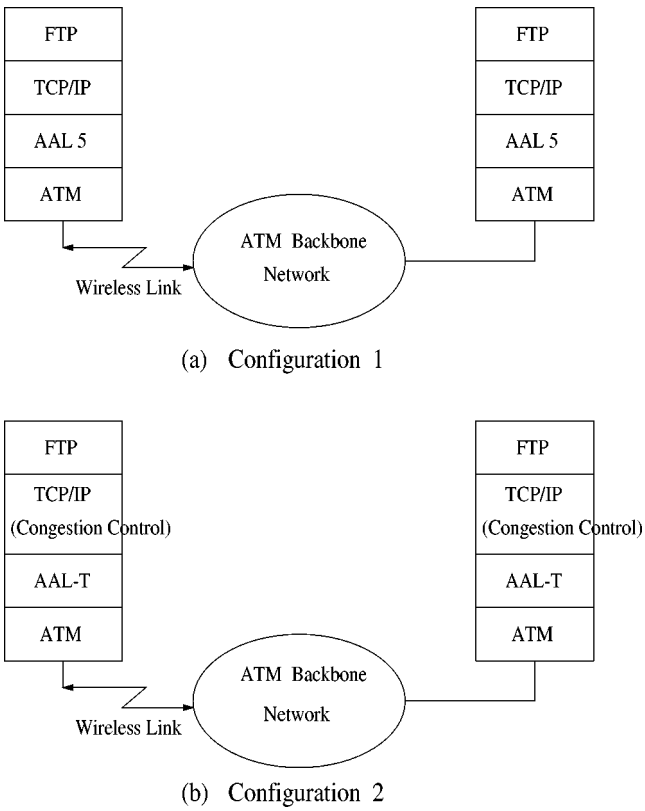

Figure 7. The network configurations.

the TCP retransmission function cannot be invoked, because timeout events never occur due to the infinite timer.

5.2. Simulation results

We consider a simulation setup for wireless ATM. We measure the following performance parameters as a function of CER on a simulated wireless ATM network:

- *TCP offered load (bits/s).* The average rate of traffic offered to the TCP layer by the applications at the source. It is calculated by dividing the total bits submitted by the simulation time.

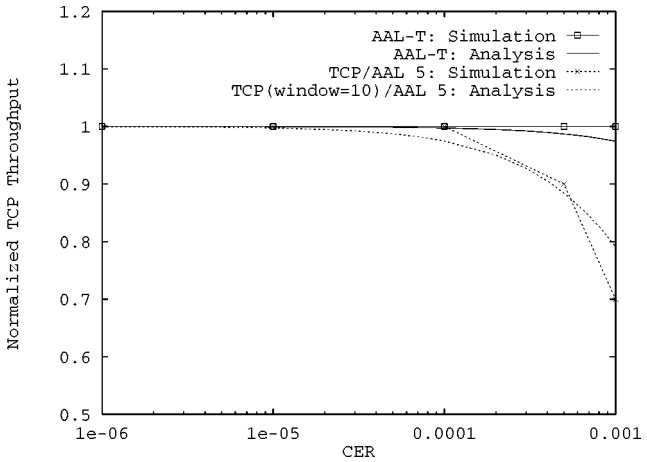

Figure 8. Comparison of normalized throughput.

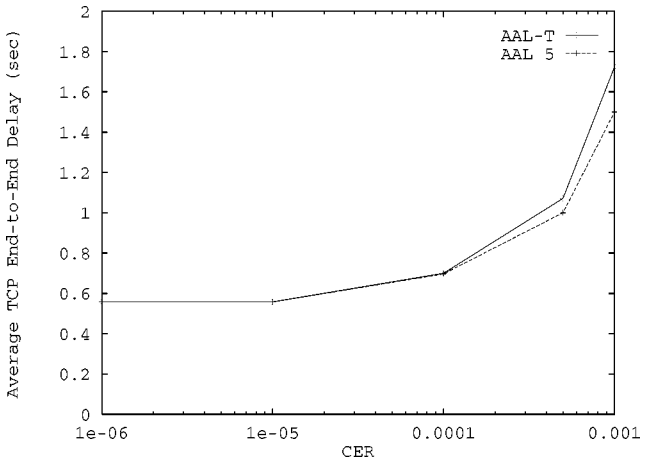

Figure 9. Comparison of average delay.

- *TCP throughput (bits/s).* The total number of bits forwarded to the application layer by the TCP layer at the destination.
- *TCP end-to-end delay time (s).* The end-to-end delay of packets received by the TCP layer. It is measured from the time an application data packet is sent to the source TCP layer to the time it is received by the TCP layer at the destination.

Figure 8 presents the normalized TCP throughput as a function of CER for AAL 5 versus AAL-T. The normalized throughput is defined as the ratio of throughput to offered load. After some transient period in the early simulation time, AAL-T keeps a perfect normalized throughput 1.0 until CER of 10^{-3} which is the range of our interest, while AAL 5 provides lower normalized throughputs of 1.0, 0.9, and 0.7 for CER of 10^{-4}, $5 \cdot 10^{-4}$, and 10^{-3} each because the TCP's GBN-ARQ scheme and congestion control mechanism are triggered in this case. If we consider that the window size of TCP is variable around 10 depending on the TCP packet size, the simulation results agree with the analytical results in figure 5(b).

On the other hand, even if AAL-T pays the price of slightly more delay due to the overhead from the CP-ARQ sublayer in return for higher throughput as shown in figure 9, this causes no problem with quality-critical, but delay-insensitive traffic such as TCP/IP traffic. For example, the average delay time of TCP in AAL-T for CER of 10^{-3} is about 1.7 s, a little longer than 1.5 s in AAL 5, but it is acceptable in case of delay-insensitive traffic. In summary, since the AAL-T provides higher throughput at the TCP level for higher CER ranges like in a wireless ATM path, it can be a good solution for improving TCP performance over wireless ATM networks.

5.3. Discussions

For single wireless segment in an end-to-end ATM path from the source to the destination, the AAL-T can be clearly justified as a better solution compared to AAL 5 from the simulation results. The AAL-T provides higher throughput for higher error ranges like in a wireless ATM path at the expense of acceptable higher delay for delay-insensitive TCP traffic.

On the other hand, let us discuss the case where the end-to-end ATM path does not contain any wireless section just like in traditional wireline ATM networks. In order to check if the AAL-T can also be applied in this case, we compare AAL 5 and AAL-T in three different perspectives. First, consider the normalized throughput at the TCP level. When the BER values of fiber links are regarded as 10^{-9} or 10^{-10}, the corresponding CER values are 10^{-6} or lower for wireline ATM networks. Therefore, there is no difference in terms of throughput at the TCP level for both AAL 5 and AAL-T over wireline ATM networks, because they provide the same normalized throughput 1.0 as shown in figure 8. Likewise, the average end-to-end delays at the TCP level are about 0.56 s for both AAL-T and AAL 5 over wireline ATM networks as shown in figure 9.

Finally, the overhead ratio is compared at the AAL level between AAL 5 and AAL-T in terms of bandwidth efficiency. Since wireline ATM networks can support the largest possible payload length (e.g., 22 Kbytes and 70 Kbytes for the BER values of 10^{-9} and 10^{-10} each) at the CP-ARQ sublayer for the best channel quality, the CPCS-PDU does not have to be segmented at the CP-ARQ sublayer and is delivered entirely by one CP-ARQ PDU. In this case, the total overhead size in AAL-T is 8 bytes (i.e., 4-byte overhead from the CPCS sublayer plus 4-byte overhead from the CP-ARQ sublayer), which is the same as in AAL 5. In summary, the AAL-T can be used for quality-critical TCP traffic over both types of ATM networks regardless of wireless or wireline, because there is no difference in terms of throughput, delay, as well as overhead ratio.

6. Conclusions

Since the poor performance of TCP over wireless ATM networks is mainly attributed to the fact that TCP always

responds to all packet losses by congestion control, the error control portion of TCP is pushed down to the AAL layer so that TCP is only responsible for congestion control. In this paper, we propose a new AAL (AAL-T) to take care of error control, which leverages off the current standard AAL 5 protocol. The AAL-T protocol is based on a reliable ARQ-based mechanism to support quality-critical TCP traffic over wireless ATM networks. Moreover, AAL-T has the ability to increase the throughput efficiency by using the variable length of CP-ARQ PDU, depending on the end-to-end path conditions. The AAL-T protocol can also improve the TCP performance under congestion conditions.

We conducted file transfer experiments for different CER values by OPNET simulation and compared with the standard approach of TCP/IP over AAL 5 in order to evaluate the TCP performance over AAL-T. The simulation results have been compared with the results from mathematical analysis. In summary, since AAL-T provides better throughput for higher CER values as compared with AAL 5, AAL-T can be a good solution for improving TCP performance over wireless ATM networks.

References

[1] I.F. Akyildiz and I. Joe, A new ARQ protocol for wireless ATM networks, in: *Proceedings of IEEE International Conference on Communications ICC '98* (June 1998) pp. 1109–1113.

[2] I.F. Akyildiz and I. Joe, TCP performance improvement over wireless ATM networks through a new AAL protocol, in: *Proceedings of IEEE GLOBECOM '98* (November 1998).

[3] ATM Forum Technical Committee, Traffic management specification (version 4.0), ATM Forum Report (April 1996).

[4] H. Balakrishnan et al., A comparison of mechanisms for improving TCP performance over wireless links, in: *Proceedings of ACM SIGCOMM '96* (August 1996) pp. 256–269.

[5] L.S. Brakmo et al., TCP Vegas: New techniques for congestion detection and avoidance, in: *Proceedings of ACM SIGCOMM '94* (August 1994) pp. 24–35.

[6] R. Caceres and L. Iftode, Improving the performance of reliable transport protocols in mobile computing environments, IEEE Journal on Selected Areas in Communications 13(5) (June 1995) 850–857.

[7] J.B. Cain and D.N. McGregor, A recommended error control architecture for ATM networks with wireless links, IEEE Journal on Selected Areas in Communications 15(1) (January 1997) 16–28.

[8] K. Fall and S. Floyd, Simulation-based comparisons of Tahoe, Reno, and SACK TCP, in: *Proceedings of ACM SIGCOMM '96* 26(3) (July 1996) pp. 5–21.

[9] ITU-T Recommendation I.363, B-ISDN ATM adaptation layer specification, CCITT SG 13 (March 1993).

[10] V. Jacobson, Congestion avoidance and control, in: *Proceedings of ACM SIGCOMM '88* (1988) pp. 314–329.

[11] V. Jacobson, Modified TCP congestion avoidance algorithm, Technical Report (April 1990).

[12] T. Kalkowski and W. Burakowski, Effectiveness of the ATM services for the TCP protocol applications, in: *Proceedings of IEEE International Conference on Communications ICC '96* (1996).

[13] M. Mathis et al., TCP selective acknowledgment (SACK) options, RFC 2018 (October 1996).

[14] K. Moldeklev and P. Gunningberg, Deadlock situations in TCP over ATM, in: *IFIP Workshop for High Speed Networks*, Vancouver, B.C., Canada (August 1994).

[15] M. Perloff and K. Reiss, Improvements to TCP performance in high-speed ATM networks, Communications of the ACM 38(2) (February 1995).

[16] J. Postel, Transmission control protocol, RFC 793 (September 1981).

[17] A. Romanow and S. Floyd, Dynamics of TCP traffic over ATM networks, IEEE Journal on Selected Areas in Communications 13(4) (May 1995) 633–641.

[18] M. Schwartz, *Telecommunication Networks: Protocols, Modeling and Analysis* (Addison-Wesley, Reading, MA, 1987).

[19] J.B. Scholz and P. Cassidy, The operation of TCP and UDP protocols over ATM radio links, Technical Report, Rome Air Development Center, NY (January 1995).

Ian F. Akyildiz received his BS, MS, and PhD degrees in computer engineering from the University of Erlangen-Nürnberg, Germany, in 1978, 1981 and 1984, respectively. Currently, he is Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology and Director of Broadband and Wireless Networking Laboratory. He has held visiting professorships at the Universidad Tecnica Federico Santa Maria, Chile, Universite Pierre et Marie Curie (Paris VI), Ecole Nationale Superieure Telecommunications in Paris, France, Universidad Politecnico de Cataluna in Barcelona, Spain, and Universidad Illes Baleares, Palma de Mallorca, Spain. He is the Editor-in-Chief of "Computer Networks" (Elsevier). He is an editor for "IEEE/ACM Transactions on Networking", "ACM-Springer Multimedia Systems", "ACM/Baltzer/URSI Wireless Networks" and "Cluster Computing". He is a past editor for "IEEE Transactions on Computers" (1992–1996) and for "Computer Networks" (1989–1999). He was the program chair of the "9th IEEE Computer Communications" workshop in 1994. He also served as the program chair for ACM/IEEE MOBICOM '96 (Mobile Computing and Networking) conference as well as for IEEE INFOCOM '98 conference. Dr. Akyildiz is an IEEE Fellow and an ACM Fellow. He received the "Don Federico Santa Maria Medal" for his services to the Universidad of Federico Santa Maria in Chile. He served as a National Lecturer for ACM from 1989 until 1998 and received the ACM Outstanding Distinguished Lecturer Award for 1994. Dr. Akyildiz received the 1997 IEEE Leonard G. Abraham Prize award for his paper entitled "Multimedia Group Synchronization Protocols for Integrated Services Architectures" published in the IEEE Journal of Selected Areas in Communications (JSAC) in January 1996. His current research interests are in wireless networks, satellite networks, ATM networks, and Next Generation Internet.
E-mail: ian@ee.gatech.edu
WWW: http://www.ee.gatech.edu/research/labs/bwn

Inwhee Joe received the B.S. and M.S. degrees in electronics engineering from Hanyang University in Seoul, Korea, the M.S. degree in electrical and computer engineering from the University of Arizona, Tucson, in 1994, and the Ph.D. degree in electrical and computer engineering from Georgia Institute of Technology, Atlanta, GA, in 1998. From 1985 to 1992, he was an Engineer in the Research Center of DACOM Corporation, where his research focused on networking software, operating systems, and distributed systems. Since 1998, he has been a Member of the Network Research Group at Oak Ridge National Laboratory, Oak Ridge, TN. His current research interests include wireless ATM networks, mobile networking, multimedia networking, and performance evaluation.
E-mail: inwhee@virtue.dsrd.ornl.gov