Journal of Digital Contents Society Vol. 10 No. 2 Jun. 2009(pp. 279-289)

무선 네트워크에서 TCP성능향상을 위한 연구

김창희*

요 약

TCP는 전송 비트 오류에 의한 패킷 손실 확률이 매우 낮은 유선망을 대상으로 설계된 프로토콜이므로, 이를 그대로 유·무선 통합 환경에 적용할 경우 TCP 송신단이 무선망의 제한된 대역폭, 높은 대기시간, 높은 비트 에러율, 임시적인 연결 끊김 등의 특성에 의해 발생하는 패킷 손실도 네트워크 혼잡에 의한 것으로 가정하여 송신단 전송률을 낮추기 때문에 성능이 저하하게 된다.

본 논문에서는 Snoop을 기반으로 BS에 Threshold 와 Lower Bound을 이용한 새로운 개선된 프로토콜을 제안한다. 이 기법은 무선 패킷 손실을 빠르게 복구하기 위하여 무선링크의 상태에 따라서 기지국의 지역 재전송 타이머를 효과적으로 조정한다. 제안한 알고리즘을 다양한 시뮬레이션을 통하여 제안한 프로토콜이 Snoop프로토콜에 비하여 연속한 패킷 손실이 발생하는 무선 링크에서 패킷 손실을 효과적으로 복구하여 TCP전송률을 향상시키는 것을 확인하였다.

A Study on Improving TCP Performance in Wireless Network

Chang Hee Kim*

Abstract

As the TCP is the protocol designed for the wired network that packet loss probability is very low, because TCP transmitter takes it for granted that the packet loss by the wireless network characteristics is occurred by the network congestion and lowers the transmitter's transmission rate, the performance is degraded. In this article, we suggest the newly improved algorithm using two parameters, the local retransmission time value and the local retransmission critical value to the BS based on the Snoop. This technique adjusts the base stations local retransmission timer effectively according to the wireless link status to recover the wireless packet loss rapidly. We checked that as a result of the suggested algorithm through various simulations, A-Snoop protocol improve more the wireless TCP transmission rate by recovering the packet loss effectively in the wireless link that the continuous packet loss occur than the Snoop protocol.

Keywords : Wireless TCP, snoop, Wireless network,

1. 서론

오늘날 인터넷에서 가장 널리 사용되는 트랜스포트 프로토콜인 TCP는 유선망을 전제로 개발, 발전해 왔다. TCP는 인터넷의 Best-Effort 특성으로 인하여 발생하는 비순서적 패킷 전달과 패킷 중복 전달, 손실 등을 보완하여 신뢰성 있게 패킷을 전송하기 위한 프로토콜이다. 네 가지(Slow Start, Congestion Avoidance, Fast Retransmit, Fast Recovery) 중요한 TCP 알고리즘은 송신자가 전송률을 조절하고 서로 다른 문제를 다룰 수 있도록 독립적으로 동작하며 RFC 2581로 표준화되었다.

하지만 패킷손실이 적은 유선망에 최적화 되어 있는 TCP를 무선망에 그대로 적용하여 사용했을 때 무선 링크의 여러 가지 원인에 의해 망의 성능 저하가 예상된다[1]. 유선링크의 낮은 BER과 랜덤하게 발생하는 에러와는 반대로 무선링크는 높은 BER과 연속적인(bursty) 패킷 유실로 인해 전송속도가 심각하게 저하된다. 또한 무선링크에서의 전송품질은 handoff와 관련된 지연 또는 연결의 손실로써 더욱 나빠질 수 있다[2]. 무선망에서의 TCP가 문제가 되는 점이 바로 이 패킷 손실 때의 혼잡제어다. 무선망의 특

※ 제일저자(First Author) : 김창희
접수일:2006년 05월 06일, 완료일:2006년 06월 12일
* 서울기독대학교 국제경영정보학과
area88@scu.ac.kr

성상 패킷 손실이 잦음에도 불구하고 TCP는 무선망에서의 패킷 손실과 혼잡에 의한 패킷 손실을 구분 할 방법이 없다. 따라서 전송 대역폭 자체가 줄어든 것이 아니라 잠시 무선망의 특성에 의해 발생한 패킷 손실을 혼잡에 의한 패킷 손실로 간주하고 혼잡제어를 하게 됨으로써 매번 패킷 손실이 일어날 때마다 송신 호스트는 보내는 속도를 낮추게 된다. 이러한 TCP의 특성은 에러가 잦은 무선망에서 TCP 성능을 크게 저하시킨다.

무선 환경에서 TCP 프로토콜의 성능저하 문제를 극복하기 위한 다양한 시도와 연구가 이루어져 오고 있다. 그들 중 대부분이 전송률을 높이기 위하여 기지국의 성능을 높이는 것에 초점을 맞추고 있다. 그 기본개념은 유선링크와 무선링크를 구분하고 기지국은 무선링크에서 일어나는 문제들을 FH쪽에 숨김으로써 TCP의 성능저하를 경감시키는 것이다.

대표적인 연결분할 방식을 취하는 방안으로는 I-TCP프로토콜이 있다[3]. Indirect TCP는 일반적인 end-to-end TCP 연결을 BS를 기준으로 하여 두 개의 분리된 TCP 연결을 설정하여 패킷을 전송하는 방법이다[4]. BS는 FH와 연결된 유선 링크와 MH와 연결된 무선 링크로 연결을 나누어 관리한다. 이 두 개의 링크는 서로 독립적으로 운영된다. BS를 기준으로 유선 링크와 무선 링크를 구분함으로써 FH와 연결된 TCP는 연결을 변경할 필요가 없고, 느린 RTT(Round Trip Time)와 재전송 패킷을 BS가 더 빨리 처리하여 각 호스트에 알릴 수 있고, FH에서는 MH가 현재의 셀에서 다른 셀로 이동을 했을 지라도 어떠한 영향을 받지 않으며, 두 개의 분리된 링크는 각각의 독립적인 흐름 제어를 할 수 있다. 그러나 I-TCP는 일반적인 TCP 개념의 종단간 전송 원칙을 위배하고 있고, BS에서는 유선 링크와 무선 링크를 각각 유지하기 위해서 충분한 각각의 버퍼를 충분히 필요로 한다. 또한 많은 수행이 꾸준히 지속되므로 큰 부하가 발생할 수도 있다. 그리고 MH의 이동에 따른 handoff 발생 시에 old BS에서 new BS로 모든 TCP 패킷들과 상태 정보를 알려줘야 하므로 handoff 발생 시에는 더 큰 latency가 발생할 수도 있다.

M-TCP는 I-TCP와 달리 일반적인 TCP 개념의 종단간 전송원칙을 유지하면서 MH를 위한 TCP 성능향상에 초점을 맞추고 있다 [5]. M-TCP는 handoff가 발생되면 MH에서 FH로 전송하는 Ack의 winddow size field를 “0”으로 셋해서 보내고, 이를 수신한 FH는 persist 모드로 유지된다. 또한 혼잡 제어로 인한 CWND를 줄이지 않음으로써 TCP의 성능 향상이 가능하게 된다. 이 방법은 실제로 handoff로 인한 TCP 연결이 끊어지는 것을 막기 위한 방법이다.

이 방법의 단점은 기본적으로 낮은 비트 에러율을 가정하여 설계되었기 때문에 FEC와 신뢰성 있는 link-layer 프로토콜로써 낮은 비트 에러율을 가질 수 있도록 하는 설계가 필요하고, MH는 handoff나 물리적 간섭 때문에 긴 연결 중단에 접할 수 있는데 이때에는 패킷의 손실로 Poor TCP throughput을 야기 시킨다.

Freeze TCP는 I-TCP처럼 split connection을 이용하지도 않으며, BS에 어떤 추가되는 사항도 없이 실제적인 TCP 세션의 종단간 전송원칙을 유지하는 방법이다[6].

Freeze TCP에서 가장 중요한 점은 MH의 이동으로 발생하는 셀 disconnection에 대한 신호를 얼마나 잘 감지하는가이다. MH는 현재 연결된 BS에 대한 신호의 강도를 무선 안테나로 모니터링하고 있으면서 임박한 handoff를 감지한다. handoff를 감지한 경우에는 MH는 Ack의 window size field를 “0”로 셋 해서 전송하고, 이를 수신한 FH는 freeze 모드로 동작하게 된다. 즉 FH는 TCP 연결의 재전송 타이머를 멈추고 CWND를 줄이는 것 없이 passive 모드로 동작하면서 세션을 유지하고 있으며, receiver window를 주기적으로 검사하기 위해서 Zero Window Probes(ZWP)를 보낸다.

Freeze TCP는 어떤 추가적인 모듈이 BS에 필요가 없으며 기존의 TCP end-to-end 연결에서의 과부하도 적다. 그러나 end host의 네크워크 프로토콜이 Ack에서는 TCP 연결이 설정된 네트워크에 대한 이동성에 대하여 인식하여야 하며, 현재 네트워크 서비스에는 Freeze TCP를 적용할 경우에는 handoff 감지가 어려우며, 이로 인한 많은 latency가 발생할 수 있다는 단점이 있다.

링크 레이어 방식 중 하나인 Snoop 프로토콜은 TCP의 종단간 특성을 유지하면서도 무선 링크에서의 패킷 손실에 의한 송신단 에서의 불필요한 TCP 혼잡제어 발생을 방지하여 TCP 전송

률을 매우 효과적으로 향상시킬 수 있다는 것이 증명되었다[7][8]. Snoop 프로토콜은 지역 재전송(local packet retransmission)을 이동 단말로부터의 중복 ACK(ack- nowledgment) 패킷의 수신이나 지역 재전송 타이머의 만료시에 의해 수행하게 된다.

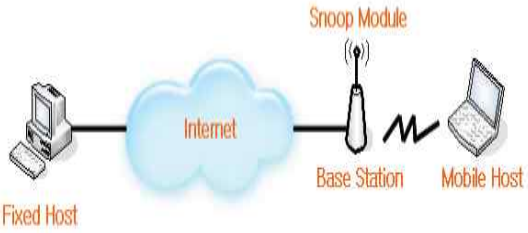
Snoop 프로토콜은 연속적인 패킷 손실이 발생하는 무선 링크에서는 ACK 패킷들이 연속적으로 손실되기 때문에 중복 ACK 패킷으로 인한 지역 재전송은 수행하기 어렵고, 주로 지역 재전송 타이머 만료에 의해 지역 재전송을 수행하게 된다. 하지만 이런 무선 링크에서는 지역 재전송 타이머 값(WRTO: Wireless Retransmission Timeout)이 커지게 되므로 무선 링크에서의 패킷 손실을 빠르게 복구할 수가 없어서 TCP 성능 저하를 초래하는 문제점이 있다.

본 논문에서는 Snoop 프로토콜을 기반으로 하면서 적응적 지역 재전송 타이머를 사용하여 이런 Snoop 프로토콜의 단점을 개선한 A-Snoop(Advanced Snoop)프로토콜을 제안하였다. 이 프로토콜은 연속적인 패킷 손실이 발생하는 환경에서 무선링크가 불량 상태에 빠진 경우, 두 개의 파라미터, 지역 재전송 시간값과 지역 재전송 임계값을 이용하여 기지국에서의 지역 재전송 시간을 아주 짧게 하여 자주 지역 재전송을 수행함으로써 기존의 Snoop 프로토콜보다 효과적으로 TCP 전송률을 향상시킬 수 있다.

본 논문은 다음과 같이 구성된다. 2장에서는 본 논문이 제안한 프로토콜의 기반이 되는 Snoop 프로토콜에 대해 알아본다. Snoop의 동작과 그 원리를 알아보고 문제점을 분석한다. 3장에서는 제안한 프로토콜을 설명한다. 4장에서는 제안한 프로토콜을 시뮬레이션 하기 위한 환경을 설명하고, 제안한 알고리즘의 파라미터 값들을 설명하고, 제안한 알고리즘의 성능을 평가한다. 다양한 시뮬레이션을 통하여 제안한 프로토콜이 기존 프로토콜보다 성능이 우수한 것을 보이고, 마지막 5장에서 전체에 대한 평가와 향후 연구되어야 할 과제를 설명한다.

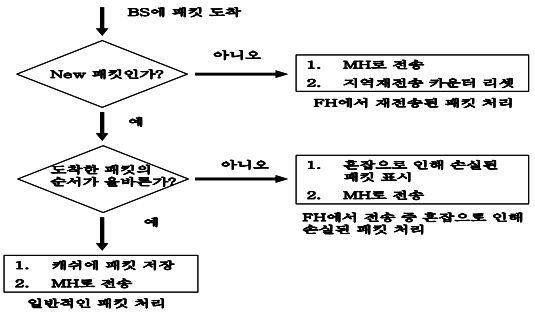
2. Snoop

Snoop은 유선망에서 사용되는 TCP의 수정 없이 사용하면서 무선망의 TCP 성능을 향상 시킨다.



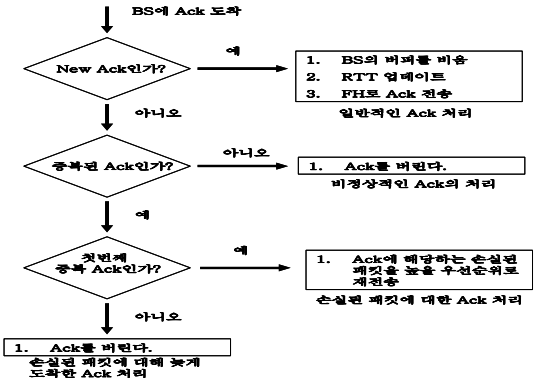
(그림 1) Snoop을 적용한 네트워크 구성도

본 장에서는 유선 환경과 무선 환경이 서로 연결되어있는 네트워크에서의 TCP 성능 개선을 위하여 현재 가장 주목 받고 있는 프로토콜이며, 제안하는 방식이 기반으로 하고 있는 Snoop 프로토콜의 작동 방식을 설명한다[7][8]. (그림 1)에서 보이는 것과 같이 Snoop은 유선망과 무선망이 혼합된 환경에서 기지국에 설치되어 작동한다. Snoop은 각 패킷 흐름에 대해 버퍼를 할당하고 버퍼에송신단으로부터 수신단 으로 전송되는 데이터 패킷을저장하며 송신단과 수신단 사이에 전송되는 ACK 패킷과 데이터 패킷을 각각의 목적지로 릴레이하는 기능을 기본적으로 구현한다. 그러나, 중복 ACK 패킷이 발생하는 경우에는 Snoop은 이것을 송신단에게 전달하지 않고, 기지국 버퍼에 저장되어 있는 패킷을 수신단에게 재전송(local retransmission)하여 패킷 손실을 지역적으로 복구함으로써 무선망에서의 패킷 손실 발생을 송신단에게 감춘다. 또한, 버퍼의 가장 앞에 있는 패킷에 대해 패킷을 전송한 시간을 시작으로 해서 일정 시간 지속되는 지역 재전송 타이머를 설정해 두고, 이 지역 재전송타이머가 만기되도록 패킷에 대한 ACK이 도착하지 않는 경우에도 무선 망 전송에서 패킷 혹은 ACK이 손실되었다고 가정하고 수신단에게 해당 패킷을 재전송한다. 데이터 패킷이 기지국의 Snoop 모듈을 통과하여 수신단에게 전달되는 과정은 (그림 2)와 같다.



(그림 2) BS가 패킷을 받았을 때의 처리과정

먼저 기지국에 데이터 패킷이 도착하게 되면, Snoop 모듈은 도착한 데이터 패킷이 새로운 패킷인가를 확인하고 새로운 패킷이 아니면 저장하지 않고 바로 수신 단에 패킷을 전달한다. 만약 도착한 패킷이 새로운 패킷이지만 순서에 맞지 않으면 기지국은 유선망에서 혼잡으로 인해 패킷손실이 발생했다고 보고 해당 패킷에 대해 혼잡을 표기한 후 수신단으로 전송한다. 그리고 도착한 패킷이 순서에 맞게 올바르게 도착한 새로운 패킷이면 복사본을 기지국의 버퍼에 저장하고 수신단에게 전달한다.



(그림 3) BS가 ACK를 받았을 때의 처리과정

(그림 3)은 TCP 수신단으로부터 ACK 패킷을 수신하였을 때 기지국에서 동작하는 Snoop 모듈의 작동 과정을 보여준다. 도착한 ACK 패킷이 새로운 ACK 패킷이면Snoop은 기지국 버퍼에 저장되어 있는 데이터 패킷들 중해당 패킷들을 버퍼에서 삭제하고, 무선 링크의 RTT(Round Trip Time)를 측정하여 이 새로운 RTT를반영하도록 지역 재전송 타이머 값을 수정한다. 도착한 ACK 패킷이 이미 버퍼에서 삭제된 패킷에 대한 ACK이라면 기지국은 수신한 ACK 패킷을 단순히 버린다. 만약 도착한 ACK 패킷이 첫번째 중복 ACK이라면 이것은 무선망의 패킷 손실을 의미하므로 기지국 버퍼에 저장되어있는 데이터 패킷들 중 손실된 데이터 패킷을 TCP 수신단에게 재전송하고, ACK 패킷은 버린다. 그러나 만약 중복 ACK 패킷이지만 첫번째 중복 ACK 패킷이 아니라면 이미 지역 재전송을 한 패킷이므로 단순히 ACK 패킷을 버린다. 이와 같이 동작하는 Snoop 프로토콜은 TCP의 종단간 연결 개념을 유지할 수 있다는 장점이 있으나, Snoop이 무선망에서 발생한 패킷 손실을 지역 재전송에 의해 복구하는 동안에 TCP의 송신단은 ACK 패킷을 기다려야 하므로 송신단 휴지 시간이 발생하여 유선망의 자원을 낭비하게 된다. 또한 반복적인 패킷 손실로 인해 기지국에서의 지역적 패킷 손실 복구에 걸리는 시간이 지나치게 길어지는 경우에는 무선망의 패킷 손실이 송신단에게 숨겨지지 못해 송신단에서 재전송 타임아웃과 불필요한 혼잡 제어가 시작될 가능성이 있다 [9].

3. 제안하는 알고리즘

TCP의 타임아웃과 재전송의 기본은 연결에 따른 RTT 측정에 있다. 이것은 경로가 바뀌고 네트워크 트래픽이 변화함에 따라서 시간도 변할 수 있어서, TCP는 항상 이 변화를 감시하고, 타임아웃 값을 적정한 값으로 변경해야 한다. TCP는 적당한 순서 번호를 가진 바이트의 송신으로부터 이의 확인 응답이 되돌아오기까지 RTT를 측정할 필요가 있다.

재전송 타이머는 네트워크 상황 변동에 따라 적응적으로 재전송 타임아웃 시간을 결정하기 위해 지속적인 RTT 측정을 수행한다. RTT는 특정 세그먼트를 전송한 시간과 해당 세그먼트에 대한 확인응답이 도착한 시간의 차를 의미한다. 그러나 상기한 바와 같이 RTT 값은 매우 가변적이기 때문에 타임아웃 시간의 갑작스런 변동을 막고, 적응력 있게 RTO를 설정하기 위해서 RFC793에 정의된 대로 exponential averaging 기법을 통해 SRTT(Smoothed Round Trip

Time)값을 다음의 식을 이용해 계산한다[10].

$$SRTT(new)=\alpha\times SRTT(old)+(1-\alpha)\times RTT(new) \quad (1)$$
$$RTO=\beta\times SRTT(new) \quad (2)$$

SRTT(new): Smoothed Round Trip Time estimate
RTT(new): current Round Trip Time estimate
α : Smoothed factor
β : delay factor

식 (1)에서 큰 α값을 사용하면 이전의 RTT 값의 비중이 커져서 새로이 측정된 RTT값에 큰 영향을 받지 않으므로 RTT 변화에 느리게 반응한다. 반대로 작은α값을 사용하면 새로 측정된 RTT값의 비중이 커져서

변화에 민감하게 반응하게 된다. 유선환경에서는 보통α의 값으로 0.875를 사용한다.

위와 같이 설정된 SRTT(new)값을 통해 식(2)를 이용해서 RTO의 값을 설정하게 된다. 일반적으로 유선환경에서는 β의 값으로 2를 사용한다[10].

[11]은 이 접근 방법의 문제점을 지적하고 있다. 이것은 불필요한 재전송에 의해 생기는 RTT의 변동에 대응할 수 없다는 것이 주요 이유이다. [11]의 지적에 따르면 이미 부하가 걸려 있을 때 불필요한 재전송은 네트워크의 부하를 더하게 된다. [12]의 연구결과에 의해 β의 값은 4로 변경 된다.

무선 네트워크 환경에서 본 논문이 제안하는 TCP 성능저하 극복방안은 Snoop 모듈을 기반으로 하되, 재전송 임계값(Threshold)과 지역 재전송 시간값(Lower Bound)를 설정하여 그 단점을 개선하였다.

기본적으로 RTT(Round Trip Time)의 측정에 의한 RTO 타이머의 설정과 이 타이머의 만기에 의한 재전송 개념은 패킷 손실 확률이 아주 낮은 유선 환경에서 개발된 개념이다.

송신 단에서는 수신 단까지의 전파시간(Propagation time)과 전송된 패킷이 중간에 경유할 수 있는 라우터의 Queuing delay와 Processing delay그리고 경유하는 링크의 전송품질을 예측할 수 없기 때문에 RTT에 의한 RTO의 설정은 유선링크상의 혼잡에 의한 패킷 손실을 효과적으로 복구하여 높은 전송률을 달성할 수 있었다.

그러나 Snoop은 다음 패킷 경로상에 라우터를 거치지 않는 링크 레이어(Link layer) 프로토콜이고 기본적으로 셀룰러 환경에서 동작하는 프로토콜이므로 Maximum Cell의 크기가 정해지므로 Maximum Propagation Delay를 예측할 수 있다. 그러므로 하나의 셀(Cell)내에서 동작하는 링크 레이어 프로토콜인 Snoop 프로토콜에서 무선 링크의 무선 WRTT (Wireless Round TripTime) 측정에 의한 무선 WRTO를 설정시에는 채널환경이 불량한 경우 연속적인 ACK 패킷과 데이터 패킷의 손실에 의해 무선 WRTT가 길어지고 결과적으로 무선 WRTO가 길어지게 된다. 따라서 기지국의 지역 재전송 간격이 길어지게 되므로 무선링크상의 패킷 손실을 빠르게 복구할 수 없다.

그러므로 Snoop 프로토콜에서 무선 WRTT 측정에 의한 무선 WRTO 설정은 비효율적이다. 따라서 제안하는 프로토콜에서는 무선채널에서 중복 ACK 패킷이나 데이터 패킷의 연속적인 손실시 발생하는 무선 WRTO 만기시간이 길어지는 단점을 개선하기 위하여 아래와 같은 알고리즘을 적용한다.

Snoop 프로토콜은 TCP와 마찬가지로 지역 재전송 타이머 값으로 RTO(Retransmission Time Out)를 사용한다. 식 (1)에서 α 값의 변화로도 다른 성능을 나타낼 수 있지만, 무선채널에서는 패킷 손실의 원인이 네트워크 혼잡에 의한 것이 아니라 높은 BER에 의해 채널 상에서 손실이 되므로 유선환경에서 사용하는 RTO 설정 알고리즘을 이용하여 기지국에서 지역 재전송을 위한무선 WRTO값을 설정하는 것은 적절하지 않다.

이 문제를 해결하기 위해 제안한 프로토콜에서는 송신단의 SRTT와 RTO 설정 알고리즘은 그대로 유지하고 지역재전송을 위해 기지국과 이동단말 간에 측정된 무선 WSRTT(Wireless Smoothed RoundTrip Time)값과 무선 WRTO값 설정 알고리즘을 무선채널의 특성을 반영하여 다음과 같이 수정한다.

$$WSRTT(new)=\alpha\times WSRTT(old)+(1-\alpha)\times WRTT(new) \quad (3)$$
$$WRTO=MAX[Lower\ Bound,\ WSRTT(new)+4\times WRTT(new)] \quad (4)$$

WSRTT(new): Wireless Smoothed Round TripTime estimate
WRTO: Wireless Retransmission Time Out
WRTT(new): Wireless Round Trip Time estimate

기지국과 모바일 단말간의 무선 WRTT가 길어지면 무선채널환경이 불량하다는 것을 의미하므로 가능하면 기지국의 무선 WRTO 간격을 좁혀서 적극적인 지역재전송을 통해 빠르게 무선상에서의 패킷 손실을 복구해야만 한다.

이것을 가능하게 하기 위해서는 유선환경과는 반대로 작은 α(=0.125)값을 사용하여 현재 무선 WRTT변화가 민감하게 반영된 무선 WSRTT값을 이용하여 빠르게 채널 환경의 좋고 나쁨을 판단해야 한다.

그러나 채널환경이 나빠지면 무선 WSRTT값도 빠르게 커짐에 따라 식 (4)에 의해서 기지국의 지역 재전송 간격도 길어져 기존 Snoop 프로토콜에서는 무선채널의 에러를 빠르게 복구할 수 없게 된다. 이것을 방지하고 무선 채널의 에러를 빠르게 복구하기 위해서 본 논문에서는 지역 재전송 임계값(Threshold)을 도입하여 무선 WSRTT가 임계값을 벗어나게 되면 채널 상태가 불량상태로 진입했다고 판단하여 식 (4)에 의한 무선 WRTO값을 무시하고 Lower Bound를 강제적으로 무선 WRTO값으로 설정하여 지역 재전송간격을 좁혀서 무선채널의 에러를 빠르게 복구할 수 있도록 한다.

채널환경이 불량상태에서 양호한 상태로 진입하게 되면 WSRTT값이 임계값 이하로 내려가므로 이때에는 위의 식 (4)에서 계산된 값을 무선 WRTO값으로 설정하여 무선 WRTO 간격을 Lower Bound보다 크게 설정함으로써 불필요한 지역 재전송을 방지하여 링크 레이어 계층에서의 트래픽을 감소시킴으로써 전송률 향상을 도모한다.

제안한 알고리즘의 의사코드(Pseudo-Code)는 다음과 같다.

```
New Ack Arrival;
WSRTT update; // WSRTT(new)=αx
        WSRTT(old)+(1- α) × WRTT(new)
IF ( WSRTT > Threshold) {
        WRTO=Lower Bound;
        //채널이 불량한 경우
ELSE
        WRTO=MAX[LowerBound,
            WSRTT(new) + 4xWRTT(new)]
        //채널이 양호한 경우;
```

WSRTT : Wireless Smoothed Round Trip Time estimate
WRTT : Wireless Round Trip Time estimate

Threshold(재전송 임계값) : 무선 링크가 양호한 상태인지, 불량한 상태인지를 판단하는 기준.

Lower Bound: (지역재전송 시간값) : 빠른 지역 재전송 위한 시간값.

제안 하는 프로토콜은 Threshold 값과 Lower Bound 값에 따라 TCP 전송률의 차이가 많이 발생하게 된다.

우선 Threshold 값은 무선 링크가 양호한 상태인지, 불량한 상태인지를 판단하는 기준이 된다.

만약 무선 링크가 불량한 상태로 판단되었다면 지역 재전송을 빠르게 수행하여 무선 링크에서 손실된 패킷을 신속히 복구해야 한다. 빠른 지역 재전송을 위해서는 미리 지정된 지역 재전송 시간값 (Lower Bound)을 WRTO 값으로 설정해 재전송을 수행한다. 이 때, Lower Bound 값은 무선 링크의 환경에 따라 다르게 설정하는 것이 필요하다. 즉, 패킷이 연속적으로 손실되는 무선 링크에서는 Lower Bound 값을 크게 하는 것이 효율적일 것이다.

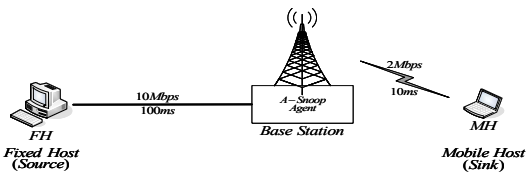
만약, Lower Bound 값을 작게 하는 경우 무선 링크가 아직 불량인 상태에서 패킷 재전송을 수행하게 되어 재전송한 패킷 또한 무선 링크에서 손실될 수 있기 때문이다. 이와는 반대로 패킷이 산발적으로 손실되는 무선 링크에서는 Lower Bound 값을 작게 하는 것이 더 효과적이라는 것을 유추해 생각해 볼 수 있다.

본 논문에서는 컴퓨터 시뮬레이션을 통해 여러 무선 링크 환경에서 더 향상된 TCP 전송률을 얻기 위한 제안하는 프로토콜의 파라미터 값들의 설정 방법에 대해 알아본다.

4. 시뮬레이션 모델 및 결과

4.1 시뮬레이션 모델

본 논문에서는 시뮬레이션과 수치해석을 위해 (그림 4)와 같이 유선 링크와 무선 링크에 각각 FH와 MH가 한 개씩 존재하는 네트워크 모델을 채용한다. 송·수신 노드가 각각 한 개인 경우 데이터 링크 계층에서의 매체 접근을 위한 경쟁이 발생하지 않기 때문에 매체에서의 패킷 충돌에 의한 패킷 손실은 발생하지 않으므로 무선 링크에서의 패킷 손실만이 TCP 전송률에 영향을 주게 된다. TCP 패킷은 FH에서 MH로만 전송된다고 가정하였고, 기지국에서는 유한한 Drop Tail 버퍼를 사용하였다. 유선 링크는 10 Mbps 대역폭과 100 ms의 전송 지연을 가지며, 무선 링크는 2 Mbps 대역폭과 10ms의 작은 전송 지연을 가진다고 가정하였다. 무선 링크에서 ACK 패킷은 손실 없이 전송된다고 가정하였는데, 이 가정은 무선 링크의 전송 지연 시간이 짧고, ACK 패킷 크기(40 bytes)가 TCP 패킷 크기(1040bytes)보다 훨씬 작은 것을 고려했을 때 타당하다[9]. 시뮬레이션 파라미터를 요약하면 <표 1>과 같다. 시뮬레이션은 무선 링크에서의 패킷 손실률(P_E)과 정규 화된 도플러 주파수(f_dT)를 변화시키면서 수행하였으며, 각기 다른 랜덤 시드(random seed)를 사용해100초 동안 10번을 수행하여 평균값을 결과 값으로 취하였다. 본 논문에서는 시뮬레이션을 위하여 버클리소재 캘리포니아 주립대학의 ns-2를 사용하였다[13].



(그림 4) 시뮬레이션 모델

<표 1> 시뮬레이션 파라미터

Parameter	Value
Maximum Segment Size	1040 Byte
ACK packet Size	40 Byte
Application Data Type	FTP
Transport Protocol	TCP-Reno
Error Model	Two-State Markov
Wired Link BER	0
Wireless Link BER(Good State)	10^{-6}
Wireless Link BER(Bad State)	10^{-2}

4.2 무선 링크 모델

무선 링크에서는 다중 경로 페이딩으로 인해 발생하는 메모리 효과 때문에 패킷 손실이 연속적으로 발생하게 된다. 따라서 유선 링크에서 사용하는 uniform 에러 모델로는 무선 링크에서 발생하는 연속적인 패킷 손실을 정확히 모델링할 수 없기 때문에, 본 논문에서는 시뮬레이션에 있어서 시간 상관성이 있는 무선 링크를 표현하기 위해 Two-state Markov 모델을 채용한다. 이 모델은 간단하면서도시간 상관성을 가지는 무선 링크를 충실히 표현할 수 있다.

무선 링크에서의 페이딩의 변화 속도는 일반적으로 정규화된 도플러 주파수(Normalized Doppler Frequency) f_dT로 표현된다. 여기서, f_d는 도플러 편이(Doppler spread)를, T는 패킷 전송에 필요한 시간을 나타내는데, 무선 링크에서 $f_dT < 0.1$ 이면 느린 페이딩(slow fading) 채널, $f_dT > 0.2$ 이면 빠른 페이딩(fast fading) 채널이라 한다[14]. 패킷 손실률이 동일한 경우, 빠른 페이딩 환경에서는 패킷 손실이 산발적으로 자주 발생하며, 느린 페이딩 환경에서 연속적으로 드문드문 발생하게 된다.

<표 2> 페이딩 변화에 따른 평균 연속손실 패킷 수 와 평균 불량시간

f_dT	P_E	Burst	MBP(sec)
0.001	0.001	12.641	0.054
	0.01	40.201	0.167
	0.1	136.56	0.568
0.01	0.001	1.4913	0.006
	0.01	4.0701	0.017
	0.1	13.671	0.057
0.1	0.001	1.0055	0.004
	0.01	1.0551	0.004
	0.1	1.5878	0.007
1	0.001	1.0011	0.004
	0.01	1.0106	0.004
	0.1	1.1168	0.005

Two-state Markov 모델을 채용하여 무선 링크를 모델링 했을 때, 페이딩의 변화 속도와 패킷 손실률에 따라 연속하여 손실되는 패킷의 수 및 평균 불량 기간을 계산하면 <표 2>와 같다. 표에서 Burst 는 패킷 손실 발생 시 연속해서 손실되는 평균 패킷의 수, MBP(Mean Bad Period)는 평균 불량 기간으로 무선 링크가 불량 상태에 진입했을 때 평균적으로 불량 상태에 머무르는 기간을 의미하며 Burst에 패킷 전송 시간 T 를 곱함으로써 구할 수 있다.

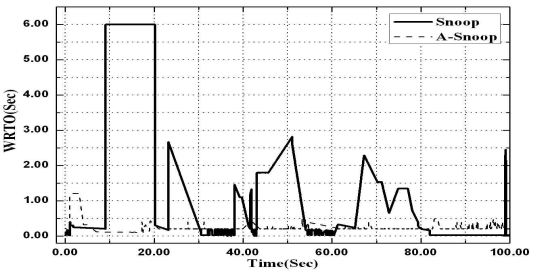
본 논문에서는 참고문헌[11]에서 사용한 계산법을 이용하여 two-state Markov 모델의 파라미터 값들을 계산하였다. <표 2>에서 보는 바와 같이 동일한 패킷 손실률을 가지는 무선 링크에서는 페이딩 속도가 느릴수록(f_dT가 작을수록) 연속해 손실되는 패킷 수가 많고 MBP 또한 컸다. 이것은 페이딩이 느리게 변화하기 때문에 한 번 무선링크가 불량 상태에 진입하는 경우 오랫동안 불량상태에 머무르기 때문이다.

이와는 반대로 페이딩속도가 빠를수록(f_dT가 클수록) Burst와 MBP는 작아졌는데, 이것은 페이딩이 빨리 변화하므로 무선링크가 불량 상태에 진입하더라도 신속히 양호한 상태로 바뀔 수 있기 때문이다. 그리고 페이딩 속도가 동일한 경우에는 패킷 손실률이 높을수록 Burst와 MBP가 모두 커져서 무선 링크에서 패킷 손실이 보다 연속적으로 발생하게 됨을 알 수 있다.

4.3 시뮬레이션 분석 및 결과

먼저 (그림 5)를 살펴보면 기존의 Snoop 프로토콜에서 무선 WRTO는 채널환경이 불량한 경우(MBP=256ms)에는 시뮬레이션의 대부분의 구간에서 크게 증가하는 것을 알 수 있다. 이런 결과는 채널환경이 불량한 경우 무선링크에서 송신단 으로 전송한 데이터 패킷과 이동 단말에서 오는 ACK 패킷이 연속적으로 손실됨에 따라 무선 WRTT값의 증가에 따른 것임을 알 수 있다.

그러나 제안한 프로토콜에서는 채널환경이 불량한 경우(MBP=256ms) 임계값을 이용한 적응력 있는 무선WRTO 설정 알고리즘을 통하여 무선 WRTO값이Lower Bound로 강제적으로 설정되어 작은 값으로 유지됨을 (그림 5)를 통하여 확인할 수 있다.

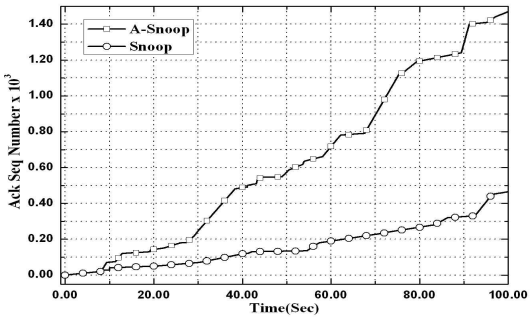


(그림 5) WRTO비교

(그림 6)은 채널환경이 불량한 경우(MBP= 256ms)에 Snoop 프로토콜과 A-Snoop 프로토콜의 성능을 비교한 그래프이다. 그림에서 세로축은 MH가 FH로 전송한 ACK패킷의 순차번호의 크기를 나타낸다.

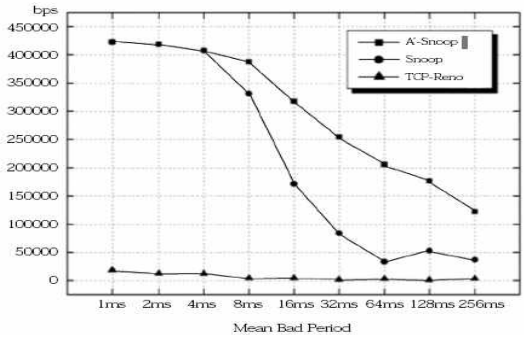
Snoop과 비교해서 A-Snoop 프로토콜에서는 증가곡선의 기울기가 갑자기 증가하는 부분이 있는데 이것은 임계값을 이용한 적응력 있는 무선 WRTO 설정 알고리즘을 통하여 무선 WRTO 간격이 좁아지게 됨에 따라 적극적인지역 재전송이 가능해져 연속적인 패킷유실로 인한 무선채널의 에러를 빠르게 복구하는 것을 나타내고 있다. 이러한 사실을 확인하기 위해서 시뮬레이션의 trace파일을 분석한 결과 Snoop의 경우

에는 중복 ACK가 연속적으로 손실되고 지역 재전송된 데이터 패킷도 채널에서 손실된 경우는 지역 재전송 간격이 길어져서 최대18초 이상 대역폭이 활용되지 못함을 확인할 수 있었고, 송신단에서 RTO의 만기가 자주 일어남을 확인하였다. 그러나 A-Snoop에서는 임계값을 이용한 적응력 있는 무선 WRTO 설정에 의해서 지역 재전송 간격이작은 값으로 일정하게 유지됨으로써 빠르게 무선채널의 패킷손실을 복구하는 것을 확인할 수 있었다.



(그림 6) Snoop과 A-Snoop의 성능비교

(그림 7)은 MBP를 1ms부터 256ms까지 2배씩 증가시켜 가며 시뮬레이션 기간 100초 동안의 Snoop과A-Snoop 프로토콜의 수신율을 나타내고 있다.



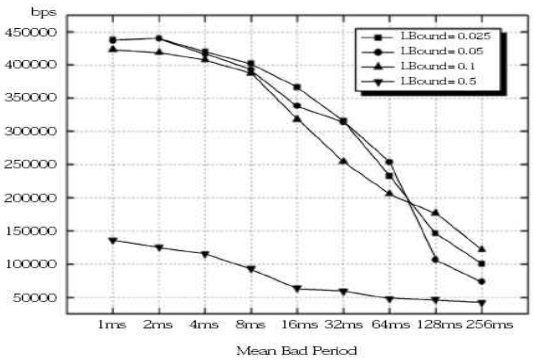
(그림 7) Snoop과 A-snoop의 수신율 비교

수신율은 시뮬레이션 기간(100초)동안 MH가 수신한 TCP데이터의 양을 시뮬레이션 기간으로 나눈 값이다.

TCP-Reno는 MBP=1ms인 경우에도 심각한 성능저하를 보인다. 그러나 Snoop과 A-Snoop 프로토콜은 MBP=8ms까지는 비슷한 성능을 보이나 16ms이후부터는 A-Snoop 프로토콜이 적극적인 지역 재전송을 통하여 Snoop 프로토콜보다 3배 이상의 성능향상을 보임을 알 수 있다.

앞의 (4)식에서 알 수 있듯이 Lower Bound값이 너무 작으면 채널 환경이 상대적으로 좋은 경우 불필요한 재전송이 자주 일어나서 Link Layer에서의 트래픽의 증가로 오히려 전송률이 하락 할 수 있다. 반대로 Lower Bound값이 너무 크면 적극적인 지역 재전송을 할 수 없어서 무선채널의 에러를 빠르게 복구 할 수 없을 것이다. Lower Bound가 A-Snoop에 미치는 영향을 알아보기 위하여 0.025, 0.05, 0.1, 0.5초 각각의 값에 대하여 MBP를 1ms부터 256ms까지 2배씩 증가시켜가며 TCP의 수신율을 비교하였다.

(그림 8)은 이것의 결과 그래프를 나타내고 있다. 예상했던 대로 Lower Bound가 0.5로 큰 경우에는 성능저하가 두드러지게 나타나며 0.025, 0.05, 0.1인 경우에는 비슷한 성능을 보임을 알 수 있다.



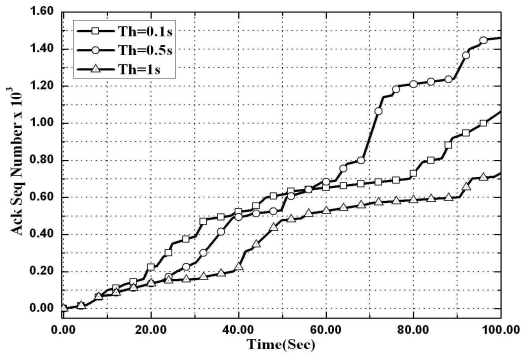
(그림 8) Lower Bound에 따른 A-Snoop 성능

A-Snoop 프로토콜에서 채널의 상태가 양호한 상태인지 불량한 상태인지를 구별하는 임계값의 설정은 성능에 아주 중요한 영향을 미친다.

임계값을 너무 크게 잡으면 채널의 변화에 민감하게 반응하지 못해서 무선채널의 에러를 효과적으로 복구할 수 없을 것이며, 반대로 임계값을 너무 낮게 잡으면 불필요한 재전송이 빈번하게 발생해서 오히려 프로토콜의 성능을 저하시킬 것이다.

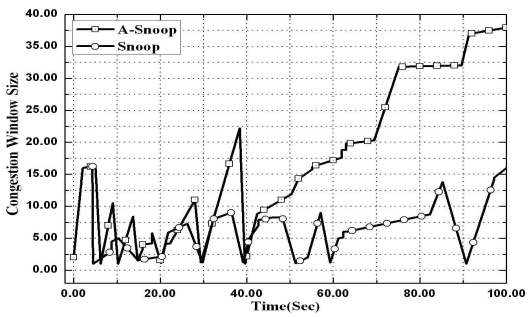
임계값이 A-Snoop 프로토콜에 미치는 영향을 알아보기 위하여 MBP=128ms와 256ms 각각의 경우에 임계값을 0.1, 0.5, 1초로 바꾸어 시뮬레이션을 수행하였다.

(그림 9)는 시뮬레이션 결과를 나타낸다. 임계값이 1초로 높은 경우에는 예상한대로 무선채널의 에러를 효과적으로 복구할 수 없는 것을 보여주고 임계값이 0.1로 낮은 경우에 무선채널의 에러를 효과적으로 복구하는 것을 알 수 있다.



(그림 9) 임계값에 따른 A-Snoop 성능비교

(그림 10)은 본 논문에서 제안한 A-Snoop과 기존의Snoop 프로토콜 각각에 대해서 채널환경이 불량한 경우(MBP=256ms)에 송신측 혼잡 윈도우(Congestion window)의 크기 변화를 보여주고 있다.



(그림 10) 송신측 혼잡윈도우의 크기 비교

그림에서 알 수 있듯이 Snoop 프로토콜의 경우에는 채널환경이 불량상태로 진입한 경우 무선 WRTO가 커짐에 따라서 적극적인 지역재전송을 하지 못해서 혼잡윈도우가 증가하다가 송신측 재전송 타이머의 만기에 의해서 혼잡제어 알고리즘이 실행되어 혼잡윈도우의 크기가 1로 떨어짐을 확인할 수 있다. 그러나 A-Snoop 프로토콜에서는 채널 환경이 불량한 상태로 진입한 경우에 적극적인 지역재전송을 통하여 Snoop 프로토콜보다 혼잡윈도우의크기가 훨씬 커지는 것을 알 수 있다. 40초 이후의 혼잡 윈도우의 크기를 비교해보면 A-Snoop 프로토콜의 적극적인 지역재전송은 송신단의 재전송 타이머 만기를 방지하여 송신단이 전체 네트워크를 혼잡상황으로 잘못 판단하여 전송률을 낮추는 혼잡제어 메커니즘의 실행을방지하는 효과도 가짐을 확인할 수 있었다.

5. 결론 및 향후 연구

Snoop 프로토콜은 기지국에서 중복 ACK 수신시에 해당 데이터 패킷을 지역적으로 재전송하여 송신측에무선망의 높은 BER로 인한 패킷 손실을 감추는 효과적인 프로토콜이다. 그러나 무선링크에서 ACK 패킷의연속적인 손실이나 기지국에서 지역적으로 재전송한 데이터 패킷의 손실 시에는 지역재전송을 위한 무선 WRTO값이 증가함에 따라서 빠르게 무선채널의 에러를 복구할 수 없는 단점이 있다.

본 논문에서는 이러한 단점을 개선하여 채널이 양호한 상태에서 불량한 상태로 진입하는 시점을 판단하기위한 임계값(Threshold)을 설정하여 기지국에서 측정한 무선 WSRTT값과 이 임계값을 서로 비교하여 무선WSRTT값이 임계값을 초과하면 채널이 불량상태로 진입했다고 판단하여 무선 WRTO값을 지역 재전송을 위한 무선 WRTO 간격의 최소값(Lower Bound)으로 강제적으로 설정하여 무선채널의 에러를 빠르게 복구할수 있는 알고리즘을 제안하였다.

또한, 시뮬레이션을 통해 제안한 A-Snoop과 Snoop 프로토콜의 성능을 비교한 결과 제안한 방법이 Snoop 프로토콜에 비하여 성능을 향상시킬 수 있음을 확인하였다. 또, A-Snoop에서 Lower Bound와 임계값에 따른 성능비교를 통하여 Lower Bound와 임계값에 적절한 값을 설정하여야 최적의 성능을 낼 수 있음을 확인하였다. 그리고 A-Snoop 프로토콜을 기지국에 적용 시는

기지국버퍼 크기를 모바일 노드 수를 고려한 시뮬레이션을 통하여 적절한 값을 적용해야 Congestion Collapse를 피하면서 높은 TCP 전송률을 달성할 수 있음을 확인하였다.

본 논문에서는 무선망에서 기존의 Snoop프로토콜보다 성능이 개선된 알고리즘을 제안하였다. 제안한 알고리즘이 기존의 방식보다 처리율을 향상시키는 것으로 나타났다. 이러한 모의실험은 실제 환경에서 다른 결과를 나타내곤 한다. 따라서 다음과 같은 연구가 향후 진행되어야 한다.

첫째, 실제 환경에서 제안한 알고리즘을 적용하여 성능향상의 결과를 확인해 보아야 한다. 이럴 경우 우리가 실험한 환경에서의 파라미터의 최적치가 달라질 수 있기 때문에 이를 고려한 파라미터들의 수정이 필요하다.

둘째, 무선 WSRTT가 커진 이유가 무선링크 에러에 의한 것이 아니라 BS 버퍼의 혼잡에 의한 상황시 적응력 있는 프로토콜이 될 수 있도록 패킷처리 알고리즘에 대한 연구가 요구된다.

마지막으로 다양한 MAC 프로토콜과 동시 사용자수의 증가에 따른 시뮬레이션 환경에서 동적인 Lower Bound와 임계값을 계산하는 연구가 필요하다. 그리고 기지국에서의 Congestion Collapse를 방지하고 높은 전송률을 달성하기 위하여 유선망의 다양한 혼잡제어 기법을 연구하여 모바일 네트워크에 적합한 혼잡제어 메커니즘을 찾는 연구가 필요하다.

참 고 문 헌

[1] Fabienne Lefevre and Guillaume vivier, "Understanding TCP's behavior over wireless links," Symposium on Communications and Vehicular technology, 2000, pp. 123-130.

[2] Hee-Jin Jang and Young-Joo Suh, "A flow control scheme for improving TCP throughput and fairness for wireless networks," IEEE Wireless Communications and Netwroking Conference (WCNC, 2003), March 2003.

[3] A. Bakre and B. R. Badrinath, I-TCP: IndirectTCP for Mobile Hosts, Proceedings of the 15th International Conference on Distributed Computing Systems, pp. 136-143, June 1995.

[4] A. Bakre, B.R. Badrinath, Handoff and System Support for Indirect TCP/IP, Second Usenix Symposium on Mobile and Location-dependent computing, Ann Arbor, Michigan April 1995.

[5] K. Brown and S. Singh, M-TCP: TCP for Mobile Cellular Networks,, ACM CCR Vol. 27(5), 1997.

[6] Tom Goff, James Moronski, Vipul Gupta, "Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments" 1995.

[7] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan and Randy H. Katz, "A comparison of mechanism for improving TCP performance over wireless links," IEEE/ACM Transactions on Networking(TON), volume 5, issue 6, page 756-769.

[8] Hari Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," ACM Wireless Networks, vol. 1, no. 4, pp. 469-481, Nov. 1995.

[9] M. Zorzi, A. Chockalingam, and A. R. Rao, "Throughput analysis of TCP on channels with memory," IEEE Journal of Selected Areas in Communications, vol. 18, no. 8, July 2000.

[10] W. Stallings, High-Speed Networks and Internets, 2nd ed., Prentice Hall, 2002.

[11] V. Jacobson, Congestion avoidance and control, Computer Communication Review 18 (4) (1988) 314 - 329.

[12] V. Jacobson, Modified TCP Congestion Avoidance Algorithm, end2end-interest mailing list, April 1990.

[13] http://www.isi.edu/nsnam/ns/

[14] A.Chockalingam and G. Bao, "Performance of TCP/RLP protocol stack on correlated fading DS-CDMA wireless links," IEEE Trans, on Vehicular Tech, vol. 49, pp. 28-33, 2003.

김 창 희

2000년 : 명지대학교 대학원 (공학석사)

2007년 : 명지대학교 대학원 (공학박사)

2008년~현재: 서울기독대학교 국제경영정보학과 교수

관심분야 : 컴퓨터 네트워크, 정보보안, 전자상거래 등