이동 애드혹 네트워크에서 AODV를 적용한 TCP에 관한 연구

A Study on TCP using AODV in Mobile Ad-hoc Networks

이혜림*, 문일영*

Hye-Rim Lee*, Il-Young Moon*

요 약

최근 네트워크 구성이 유선 환경에서 무선 환경 및 모바일 환경으로 확대됨에 따라 다양한 환경에 최적화된 다수의 TCP 알고리즘들이 제시되고 있다. 그러나 TCP는 혼잡에 의해 패킷 손실이 발생하는 유선 링크를 기반으로 설계되었기 때문에 상대적으로 유선 링크에 비해 전송 에러(Transmission Error)가 큰 무선 링크의 경우 TCP의 성능은 많이 저하된다. 본 논문에서는 이동 ad-hoc 네트워크에서 TCP의 혼잡제어 알고리즘이 얼마나 영향을 미치는지 알아보고, TCP-Tahoe, TCP-Reno, TCP New-Reno 및 TCP-Vegas가 이동 ad-hoc 환경에서 가지는 성능을 비교하였다.

Abstract

Recently, network component become to follow wireless and mobile network from wired network proposed that many TCP algorithm optimized in variety environment. When TCP was created, however as it was design based on wired link, wireless link made more transmission error than wired link. Transmission errors are more frequent and may be incorrectly regarded as indications of network congestion. In this paper, it conduct how effect congestion control algorithm in Mobile ad-hoc network and compare traffic of TCP-Tahoe, TCP-Reno, TCP-New-Reno and TCP-Vegas in Mobile ad-hoc environment.

Key words : TCP, AODV, NS-2, Mobile Ad-hoc Network

I. 서 론

최근 유비쿼터스 시대를 맞아 무선을 이용한 인터넷 사용이 점차 증가하고 있다. 또한 센서 네트워크는 유비쿼터스의 발전을 위한 핵심적인 기술로 대두되고 있다. 센서 네트워크의 특징은 움직이는 노드들이 서로 통신을 하거나 임의로 망을 만들어서 통신을 하는 ad-hoc 환경이라는 것이다. 이에 따라 이동 단말기 간의 멀티 홉을 사용하여 송신자와 수신자사이의 데이터 전송을 가능하게 하는 ad-hoc 네트워크에 대한 연구가 활발히 진행되고 있다. ad-hoc 네트워크는 유선망과 달리 무선 매체를 사용하기 때문에 경로손실, 페이딩, 노이즈, 간섭, 핸드오프, 히든 터미널 문제 등으로 높은 BER(Bit Error Rate)를 가지며, 노드들의 이동성으로 인해 동적인 토폴로지를 구성한다[1]. 하지만 유선 링크를 기반으로 설계된 TCP알고리즘은 모든 손실이 망의 혼잡으로 인식하기 때문에 전송에러에 의한 손실도 혼잡 손실로 인식하여 혼잡 제어 알고리즘(Congestion Control Algorithm)을 실행하게 된다[2],[3]. 이러한 TCP동작은 혼잡 손실이 아닌 패

* 한국기술교육대학교 인터넷미디어공학부(School of Internet Media Eng., Korea University of Technology and Education)
· 제1저자 (First Author) : 이혜림
· 접수일자 :2008년 5월 20일

킷 손실로 발생할 경우, 상당한 성능 저하를 초래하게 된다[4]. 따라서 본 논문에서는 Ⅱ장에서 TCP 혼잡 제어 방법에 대해 살펴보고, Ⅲ장에서 이동 ad-hoc 환경에서의 TCP알고리즘이 가지는 문제점에 대해 알고 본 뒤 Ⅳ장에서 AODV(Ad-hoc On-demand Distance Vector)를 적용한 ad-hoc 네트워크에서 각 TCP알고리즘이 가지는 성능을 분석을 하고 이를 통해 가장 효율성 높은 TCP알고리즘 평가 한 후, Ⅴ장에서 본 논문의 결론과 추후 필요한 연구에 대해 기술한다.

Ⅱ. TCP의 혼잡제어기법

TCP는 흐름 제어와 혼잡 제어를 사용하여 패킷의 흐름이 지연과 대역폭 등의 네트워크 상황에 따라 유연하게 적응할 수 있도록 데이터 트래픽을 제어한다[5]. 다음은 TCP-Tahoe, TCP-Reno, TCP New-Reno, TCP-Vegas의 각 TCP 버전별로 혼잡 제어 메카니즘의 차이점을 설명하고자 한다.

2-1 TCP-Tahoe

송신자가 패킷을 전송한 후 일정 시간 동안 수신자에게 응답이 없으면 망 내에서 혼잡이 발생하였다고 판단하여 임계 윈도우 값(ssthresh)을 현재 임계 윈도우 값의 1/2로 setup하고, 혼잡 윈도우 크기를 1로 setup하여 이용하는 방식이다.

2-2 TCP-Reno

TCP-Reno는 슬로우 스타트, 혼잡 회피, 빠른 재전송, 빠른 복구의 네 개의 알고리즘으로 이루어져 있다. TCP-Tahoe를 향상시킨 TCP로써, 빠른 복구를 사용함으로 TCP-Tahoe에 비해 패킷 손실 후 빠르게 전송률을 회복하는 것이 가능하다. 그러나 이처럼 TCP-Reno의 빠른 회복은 한 혼잡 윈도우 안에 한 패킷이 손실될 경우에 대해 최적화 되어있고, 한 RTT(Round Trip Time)당 손실된 패킷 하나만 재전송할 수 있기 때문에 한 혼잡 윈도우 안에서 여러 개의 손실된 패킷이 존재 할 경우 성능이 많이 저하된다는 단점을 갖고 있다.

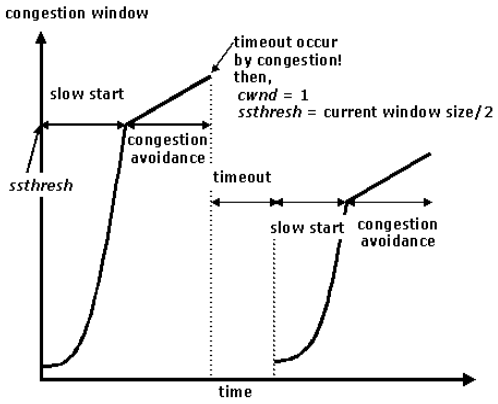


그림 1. TCP-Tahoe의 혼잡 제어
Fig. 1. The congestion control of TCP-Tahoe

2-3 TCP New-Reno

TCP New-Reno는 한 혼잡 윈도우 안에서 두 개 이상의 패킷이 손실될 경우 재전송 타임아웃에 의해 패킷을 재전송하는 TCP-Reno의 단점을 보완하였다. 이 기법은 윈도우 당 한 개 이상의 패킷 유실시 각각의 유실된 패킷의 재전송 타임아웃을 기다리지 않고 한 RTT당 한 개씩 유실된 패킷을 재전송 하도록 설계되었다. TCP New-Reno에서는 손실된 패킷을 재전송하기 전에 마지막으로 전송한 패킷의 시퀀스보다 작은 시퀀스를 갖는 패킷에 대한 ACK인 Partial ACK를 이용하여 빠른 회복의 종료여부를 결정한다.

2-4 TCP-Vegas

TCP-Vegas는 종단간 경로에서 전송중인 세그먼트의 수를 파악하여, 이로 부터 혼잡 상황을 예측한다[6]. 송신측에서 전송을 했지만 아직 수신측까지 도달하지 못한 패킷이 많다면 이는 네트워크의 버퍼에 전송대기 중인 패킷이 많다는 것으로 해석하여 혼잡 상황이라고 인식한다. TCP-Vegas에서 송신측의 RTT마다 전송중인 세그먼트 수를 계산하여 이 값이 기준값보다 큰 경우 혼잡 윈도우를 한 세그먼트만큼 증가시키고, 작은 경우 한 세그먼트만 감소시켜 네트워크의 가용 대역폭을 효율적으로 사용한다. TCP-Vegas는 현재 많이 사용되고 있는 TCP-Reno에 비해 높은

성능을 보이지만, RTT마다 변화하는 혼잡 윈도우의 크기가 세그먼트 한 개로 제한되므로 쓸 수 있는 대역폭의 변화가 클 경우 잘 적응하지 못하는 단점이 있다.

Ⅲ. 이동 애드혹 환경에서의 TCP

이동 ad-hoc 네트워크에서는 기존의 유선네트워크에 비해 데이터를 효율적으로 전송하는데에 많은 어려움이 있다. 이동 ad-hoc 네트워크는 유선망과 달리 무선 매체를 사용하기 때문에 경로손실, 페이딩, 노이즈, 간섭, 핸드오프, 히든 터미널 문제 등으로 높은 BER을 가지며, 노드들의 이동성으로 인해 동적인 토폴로지를 구성한다. 더욱이 TCP는 이러한 특징으로 발생하는 패킷 손실을 네트워크상의 혼잡으로 해석하여 혼잡제어 메커니즘을 실행하게 되고 이는 심각한 성능저하를 불러일으킨다.

앞서 말한 것처럼 이동 ad-hoc 네트워크는 스스로 자가 망을 구성하는 노드들로 구성되어 있다. 또한 이동 ad-hoc 네트워크 환경에서 각 노드들은 이동하기 때문에 출발지에서 목적지로 가는 경로는 자주 변화하게 된다. ad-hoc에서 사용하는 라우팅 프로토콜은 필요시 경로를 탐색하여 경로를 재설정하는 On-demand 방식의 라우팅 프로토콜을 사용한다. 따라서 출발지와 목적지 사이의 여러 경로를 가지고 있고 이로 인해 수신측에서 받은 패킷들의 순서 번호가 순차적이지 않다. 패킷들이 비순차적으로 수신될 경우 TCP는 duplicate ACK를 발생하고, 송신측에서 duplicate ACK 3개를 받으면 혼잡윈도우의 크기를 1로 줄이고 혼잡제어를 실행시키므로 TCP의 성능이 떨어지게 된다.

Ⅳ. 시뮬레이션 및 결과 분석

본 장에서는 기존의 TCP알고리즘에 대해서 버클리 소재 캘리포니아 주립대학의 NS-2 (Network Simulator 2)[7]를 통하여 TCP 버전 별로 알고리즘의 비교 및 성능 평가를 수행 하였다.

4-1 시뮬레이션 환경

기본적으로 링크계층은 IEEE 802.11 MAC 프로토콜 표준을 사용하였으며[8], 대역폭은 15Mbps이고 전송 지연 시간은 10ms로 설정하였다. 아래 그림2와 같이 7개의 이동 노드를 이용하고, 900M*900M이내의 토폴로지가 구성되며 평탄한 경우라고 지정한다. 라우팅 프로토콜은 AODV를 사용하여 각각 TCP알고리즘의 성능에 미치는 영향을 비교하였다.

그림2와 같이 시뮬레이션의 시작 시간은 10초로 설정하였고 노드1은 싱크노드인 노드 7에게 데이터를 전송한다. 이때 거치게 되는 홉 수는 6개이다. 50초 후에 노드5가 노드4와 6의 범위 밖으로 벗어나서 경로가 단절된다. 90초부터 노드 7이 이동하여 4홉을 거쳐 데이터가 전송되고, 120초가 되면 노드 7은 노드 2의 근처로 다가가 2홉만 거쳐 데이터를 수신하게 된다. 또한 시뮬레이션은 종료는 150초이다.

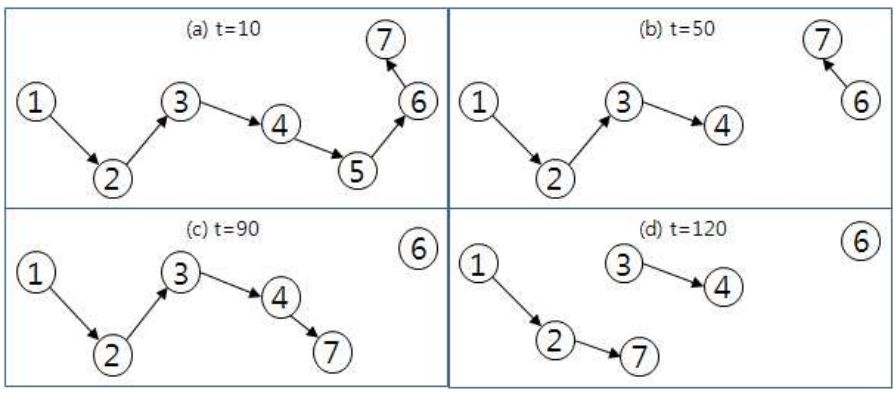


그림 3. 시뮬레이션 모델
Fig. 2. The simulation model

4-2 시뮬레이션 결과

이동 ad-hoc 환경에서 전송 계층 프로토콜로 TCP-Tahoe 와 TCP-Reno, TCP New-Reno 및 TCP-Vegas를 채택하였을 때, ACK 패킷들의 순차 번호는 그림 3과 같다.

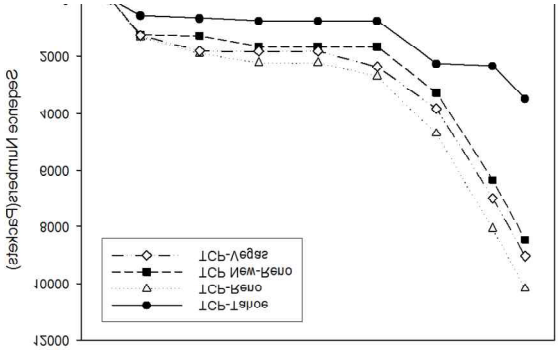


그림 3. TCP 패킷의 순차번호
Fig. 3. The sequence number of TCP packet

이동 ad-hoc 네트워크는 경로가 일정하지 않으며, 토폴로지 또한 동적으로 구성되는 특징이 있다. 그림 2 (b)에서 50초일 때 노드5의 움직임으로 경로가 단절되고 그에 따라 각 TCP들의 순차 번호는 증가 하지 않는다. 90초에 노드7의 움직임으로 경로가 재설정 됐지만, TCP-Tahoe의 경우는 경로가 재설정돼도 순차번호가 바로 증가 하지 않는다. 이것은 재전송된 패킷에 대한 ACK가 도착 할 때 까지 다른 패킷을 보낼 수 없게 되므로 경로가 재설정되어도 일정시간 동안 순차번호가 수평을 그리고 있는 것이다.

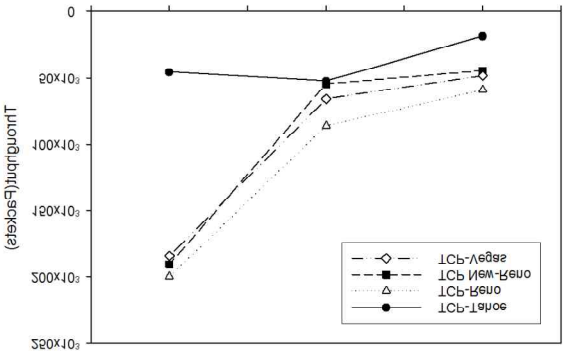


그림 4. 홉 수에 따른 전송률
Fig. 4. Throughput of hop count

또한 TCP-Tahoe를 제외한 다른 TCP들은 10초에서 50초까지 완만한 수평선을 그리고 있는 것에 반해, 120초에서 150까지는 급격한 상승곡선을 그리고 있다. 이것은 데이터를 전송하기 위해 거치는 홉 수가 높을수록 데이터의 순차번호가 상승곡선이 아닌 수평을 그리고 있다는 것을 알 수 있다.

이동 ad-hoc 네트워크 환경은 노드들이 스스로 자가 망을 구축하므로 멀티 홉 환경을 가진다. 그림 4는 홉 수가 6개, 4개, 2개 일 때 TCP의 전송률을 나타낸 것이다. 홉 수가 높을수록 TCP의 전송률이 낮아지는 것을 알 수 있는데, 특히 TCP-Tahoe는 멀티 홉 통신을 할 때 전송률이 50000packets 정도로 가장 낮다. 반면에 TCP-Reno는 멀티 홉 통신을 할 때 최고 200000packets정도로 가장 높은 Throughput을 가진다.

그림 5는 각 TCP별 혼잡 윈도우의 크기 변화를 나타낸다.

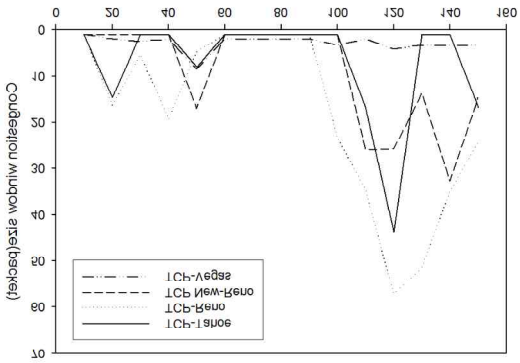
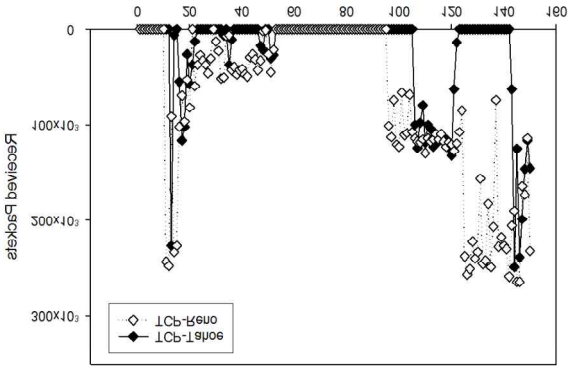


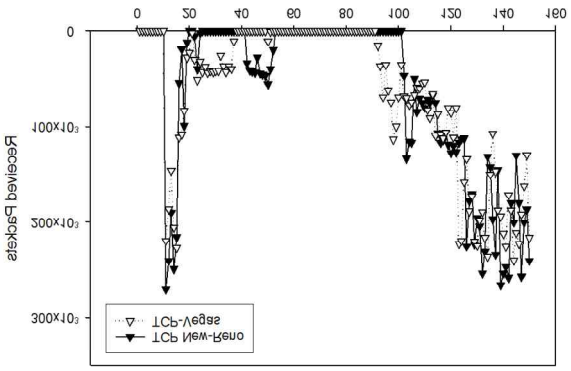
그림 5. 각 TCP의 혼잡 윈도우 크기
Fig. 5. Congestion window size of each TCP

이중 TCP-Vegas의 혼잡 윈도우의 크기는 TCP-Vegas에서 전송이 가능한 세그먼트의 양을 계산하여 그 기준 값에 따라 혼잡 윈도우의 크기를 한 세그먼트만큼 감소하거나 증가시키기 때문에 혼잡윈도우의 크기의 변화가 작고, 이 때문에 네트워크의 대역폭을 효율적으로 쓸수 있다. 또한 TCP-Vegas를 제외한 각 그림은 이동노드의 움직임에 의해서 경로가 단절되어 아무런 데이터의 송수신이 없는 50초 이후부터 TCP는 임계 윈도우 값을 현재 값의 반으로 맞추고 혼잡윈도우는 1로 설정하여 저속 시작단계로 들어가게 된다. 또한 경로가 재설정되어 데이터를 주고받을 수 있는 상황이 되면 혼잡윈도우는 1부터 지수적인 증가를 하게 된다. 이것은 TCP가 이동노드의 움직임에 대한 경로 단절을 혼잡 상황으로 인식하여 혼잡 제어 알고리즘을 실행한 결과라고 할 수 있다.

실제로 이동 ad-hoc 네트워크는 노드들의 움직임이 많기 때문에 이런 상황이 자주 발생한다. 이러한 적응력 없는 TCP는 혼잡 제어 알고리즘을 불필요하게 발생하고, ad-hoc 네트워크의 성능을 저하시키는 원인이 된다.



(a) TCP-Tahoe와 TCP-Reno



(b) TCP New-Reno와 TCP-Vegas

그림 6. 각 TCP의 전송률

Fig. 6. Throughput of each TCP

그림 6(a)는 TCP-Tahoe와 TCP-Reno의 Throughput을 나타낸 것이다. TCP-Tahoe는 홉 수가 6개 일 때 낮은 전송률을 보여주고 있으며, 이동 노드로 인해 경로가 재설정되어도 노드들의 움직임으로 인해 패킷들의 손실이 많아지고, 이 손실된 패킷들은 재전송된다. 재전송 하는 과정에 있어서 TCP-Tahoe는 재전송 타이머와 상관없이 바로 해당 패킷을 전송하고 저속 시작으로 들어가기 때문에 비효율적이다. TCP-Reno는 홉 수가 높아질수록 높은 성능을 보이고, 4개와 2개의 홉을 거치는 90초 이후부터는 TCP-Tahoe, TCP New-Reno와 TCP-Vegas보다 평균 200000packets을 받으면서 상당한 성능 차이를 보이고 있다.

그림 6(b)는 TCP New-Reno와 TCP-Vegas의 Throughput을 나타냈다. TCP New-Reno는 시작할 때 좋은 성능을 보여주다가 이후에 여러 손실이 발생하면서 급격히 성능이 저하된다. 또한 TCP New-Reno는 많은 수의 패킷손실이 발생할 경우 많은 수의 RTT가 발생하기 때문에 이동 ad-hoc 네트워크에 적용하지 못하는 부적당한 TCP이라는 것을 확인하였다. 그리고 TCP-Vegas는 전송률을 미리 예측함으로써 망의 혼잡을 피하는 알고리즘이기 때문에 다양한 원인으로 패킷 손실이 발생하는 이동 ad-hoc 환경에서는 TCP-Reno보다 조금 낮은 Throughput을 가진다. 두 알고리즘은 200000packets에 못 미치는 패킷 전송률을 보이고 있으며, 이는 보통 200000packets을 받는 TCP-Reno보다 낮은 성능을 가진다.

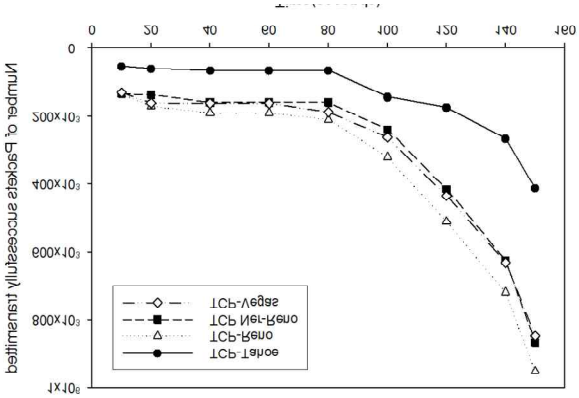


그림 7. 성공적으로 전달된 누적 패킷 수

Fig. 7. Cumulative packet number of packets successfully transmitted

그림 7은 그림 6의 전송률의 변화에 대해 성공적으로 전달된 누적패킷수를 나타낸 그림이다. 그림 6에서 TCP-Reno는 다른 TCP에 비해 눈에 띄게 성능이 좋지는 않았지만, 여러 홉을 거치면서도 안정적으로 데이터의 전송이 이루어지는 것을 알 수 있다. 이를 바탕으로 살펴본 누적패킷수도 TCP-Reno가 1000000packets에 근접하여 다른 TCP에 비해 높다는 것을 알 수 있고, 이것은 TCP-Reno가 멀티 홉 통신을 하는 이동 ad-hoc 네트워크에서 가장 안정된 연결을 하는 것을 알 수 있다.

Ⅴ. 결 론

본 논문에서는 이동 ad-hoc 네트워크에서 혼잡제어에 의한 각 TCP알고리즘 별로 트래픽 성능을 분석해 보았다. 혼잡제어 알고리즘이 멀티 홉을 가진 이동 ad-hoc 네트워크 환경에서 얼마만큼 영향을 미치는지 알아보았고, 대표적인 방식들인 TCP-Tahoe, TCP-Reno, TCP New-Reno 및 TCP-Vegas를 동일한 상황에서 모의 실험하였다. 또한, 이동 ad-hoc 환경에서 가장 빈번하게 일어나는 상황인 노드들이 이동할 때와 거치는 홉 수에 따라 변하는 각 TCP에 대한 ACK 패킷의 순차번호와 혼잡윈도우크기 그리고 Throughput을 알아보았다. 위에서 보인 모의실험을 통해 무선 환경에서 제일 좋은 성능을 보인 알고리즘은 TCP-Reno이다. TCP-Reno는 이동 ad-hoc 환경에서 안정적으로 패킷을 수신하였고, 멀티 홉 통신에서도 다른 TCP알고리즘에 비해 높은 Throughput을 보였다. 하지만 무선 환경은 제약이 많고 히든 터미널 문제 등의 여러 문제들이 고려되어야 한다. 패킷 손실을 혼잡 상황으로 인식하지 않기 위해 크로스 레이어 개념을 도입하여 하위 계층에서 전송 계층으로의 정보 전달을 이용한 효율적인 TCP알고리즘을 연구 해야 할 것이다. 따라서 많은 연구를 통해 신뢰성을 유지하면서 혼잡상황을 정확하게 이해할 수 있는 알고리즘을 지속적으로 연구해야 한다.

참 고 문 헌

[1] G. Xylomenos and G. C. Polyzos, "TCP Performance Issues over Wireless Link," *IEEE Communications Magazine*, vol. 39, pp.52-58, April 2001.

[2] W. Stevens, "*TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms*", RFC-2001, Jan. 1997.

[3] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno, and SACK TCP," *Comput. Commun. Rev.*, 1996.

[4] C. Barakat, E. Altman, and W. Dabbous, "On TCP Performance in a Heterogeneous Network: A Survey," *IEEE Communications Magazine*, Jan. 2000.

[5] 박원서, 김범준, 이재용 외 2인, "위성링크에서 혼잡한 전송에러에 의한 패킷 손실 구분을 통한 TCP 성능 개선 방안", *SK Telecommunication Review 제 10권 6호*, 2000.

[6] Lawrence S. Brakmo, San W. O'Malley, and Larry. L. Peterson. TCP Vegas: New techniques for congestion detection and avoidance, *In Proceedings of ACM SIGCOMM '94*, pages 24-35, May 1994.

[7] The Network Simulator ns-2, http://www.isi.edu/nsnam/ns.

[8] IEEE 802.11 working group, "IEEE Std 802.11, 1999 Edition," available from http://standards.ieee.org/catalog/olis/laman.html, ISO/IEC 8802-11: 1999.

이 혜 림 (李憓f琳)

2005년 3월 ~ 현재 : 한국기술교육대학교 인터넷미디어공학부 재학
관심분야 : 무선 TCP, 무선 메쉬 네트워크, 라우팅 프로토콜

문 일 영 (文日永)

2000년 2월 : 한국항공대학교 항공통신정보공학과 (공학사)
2002년 2월 : 한국항공대학교 대학원 항공통신정보공학과 졸업(공학석사)
2005년 2월 : 한국항공대학교 대학원 정보통신공학과 졸업(공학박사)
2004년 ~ 2005년 : 한국정보문화진흥원 선임연구원
2005년 3월 ~ 현재 : 한국기술교육대학교 인터넷미디어공학부 조교수
관심분야 :무선 인터넷 응용, 무선 인터넷, 모바일 IP