

유무선 복합망에서 Acknowledgement 패킷의 분할을 통한 TCP 프로토콜의 성능향상 기법

진 교 홍[†]

요 약

본 논문은 최근 급속히 증가되고 있는 무선 인터넷에서 발생하는 짧은 TCP 트래픽 응용 서비스의 성능을 향상시키기 위하여, TCP 프로토콜의 폭주제어 알고리즘을 보완하는 응답패킷 분할 기법(SPACK)을 제안하였다. 유선통신 환경과는 달리 무선통신 환경에서는 높은 비트 오류율로 인하여 TCP 프로토콜의 폭주제어 알고리즘이 오동작을 일으키게 된다. 이로 인하여 TCP 프로토콜의 성능은 급격히 저하되어 전체적인 인터넷 서비스의 성능이 떨어지게 된다. 본 논문에서는 무선통신환경에서 TCP 프로토콜의 성능을 개선시키기 위해 기지국에서 응답패킷을 분할하여 전달하는 SPACK 기법을 제안하였다. 제안된 기법은 컴퓨터 시뮬레이션을 통하여 성능을 분석하였으며, 그 결과 기존의 TCP 프로토콜에 비하여 SPACK을 이용하는 경우 더 높은 성능이 발휘됨을 확인하였다.

Performance Improvement Method of TCP Protocol using Splitting Acknowledgement Packet in Integrated Wired-Wireless Network

Kyohong Jin[†]

ABSTRACT

In this paper, in order to improve the performance of TCP short traffic application services in wireless Internet environments, the Split-ACKs (SPACK) scheme is proposed. In wireless networks, unlike wired networks, packet losses will occur more often due to high bit error rates. Therefore, each packet loss over wireless links results in congestion control procedure of TCP being invoked at the source. This causes severe end-to-end performance degradation of TCP. In this paper, to alleviate the TCP performance, the SPACK method, split acknowledgement packets in the base station, is proposed. Using computer simulation, the performance of TCP using SPACK is analyzed and shows better performance than traditional TCP protocol.

키워드 : TCP 프로토콜의 성능향상 기법(Performance Improvement Method of TCP protocol), 폭주제어 알고리즘(Congestion Control Algorithm), 유무선복합망(Integrated Wired-Wireless Network), SPACK

1. 서 론

1990년대 초반에 약 150,000명에 불과하였던 인터넷 사용자의 수는 웹(Web)의 등장과 함께 매년 폭발적인 증가 추세를 보이고 있으며, 국내만 하더라도 전자우편, 웹 등과 같은 인터넷 서비스의 사용자 수가 매달 90만명씩 늘어 현재는 약 2,300만명에 이르고 있다[1].

한편, 인터넷 트래픽은 TCP, UDP 및 기타 프로토콜용 트래픽으로 분류되며 이들을 이용하는 응용프로토콜 중 HTTP용 트래픽이 전체 트래픽의 70%로 가장 많은 부분을 차지하고 기타 DNS, SMTP, FTP, NNTP 등의 트래픽은 5%이내의 값을 보이는 것으로 나타났다[2]. 또한 이러한 용

용프로토콜이 TCP, UDP 프로토콜 중 어느 프로토콜을 많이 사용하는지를 조사해본 결과 TCP 트래픽이 전체의 90%를 차지하고 UDP는 10%정도이며 기타 ICMP 등의 프로토콜은 2%이하로 나타났다. 즉 TCP 트래픽의 대부분은 HTTP 프로토콜용 트래픽이 차지하고 있으며, 이를 HTTP용 서버와 클라이언트로 나누어 보면 클라이언트는 한 흐름(Flow)당 15개의 패킷이 발생되며, 패킷의 평균 크기는 70바이트 정도이다. 그리고 서버는 한 흐름당 15개의 패킷을 생성하고 패킷의 평균 크기는 10K 바이트이다. 또한 전체 HTTP 트래픽에서 클라이언트는 15%, 서버는 85% 정도를 차지한다[2]. 따라서 인터넷 트래픽의 대부분은 10K 바이트 길이의 짧은 패킷들이 차지하고 있다.

한편, 최근 들어 노트북, 랩탑, PDA 및 셀룰러 이동전화 등과 같은 이동단말기의 지속적인 보급으로 인해 언제, 어

[†] 정 회 원 : 동의대학교 멀티미디어공학과 교수
논문접수 : 2001년 10월 23일, 심사완료 : 2001년 12월 11일

디서나 인터넷 서비스를 이용하고자 하는 사용자가 날로 증가되고 있다. 한국인터넷정보센터의 자료에 의하면 2001년 7월 현재 국내의 무선인터넷 가입자가 2,150만명을 넘어선 것으로 발표되었다[3].

이러한 무선 인터넷 서비스를 위해서는 무선망이 뒷받침되어야 하는데, 무선망은 기존의 유선망과는 달리 신호의 페이딩이나 간섭현상으로 인하여 유선링크에 비해 비트 오류율이 높다. 즉 비트 오류율이 $10^{-3} \sim 10^{-5}$ 으로 기존 유선링크의 $10^{-6} \sim 10^{-12}$ 에 비해 상당히 높으며, 이로 인하여 패킷 손실이 빈번히 발생될 수 있다. 또한 셀룰러 망에서는 이동 단말기가 한 셀에서 다른 셀로 이동할 경우 새로운 주파수 채널을 할당하는 핸드오프(Handoff) 과정을 수행하게 된다. 이 핸드오프는 수십 msec에서 길게는 수초가 걸리는 작업이므로 일시적으로 연결이 단절될 수 있다. 따라서 무선망은 높은 비트 오류율과 연결단절과 같은 열악한 환경으로 인하여 유선망에서와는 달리 패킷손실이 자주 발생된다.

이러한 높은 패킷 손실율로 인하여 대부분의 인터넷 서비스에서 사용되는 TCP 프로토콜의 폭주제어(Congestion Control) 알고리즘은 오동작을 일으키게 되어 인터넷 서비스의 성능을 저하시킨다.

따라서 본 논문에서는 무선 인터넷 환경에서 10KB 정도의 짧은 트래픽용 서비스의 성능을 향상시킬 수 있는 기법을 제안하고 성능을 분석하였다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 무선환경에서 TCP 프로토콜의 문제점을 제시하였으며, 3장에서는 기존의 연구결과를 비교분석하고 4장에서는 제안된 기법을 설명하였다. 그리고 5장에서는 성능분석 결과를 제시하였고, 마지막으로 6장에는 결론을 기술하였다.

2. TCP 프로토콜의 폭주제어 알고리즘

현재 TCP 프로토콜로 널리 사용되고 있는 4.3BSD의 TCP-Reno버전은 Slow Start, Congestion Avoidance, 및 Fast Retransmit/Recovery 알고리즘 등을 이용하여 폭주를 제어한다[4].

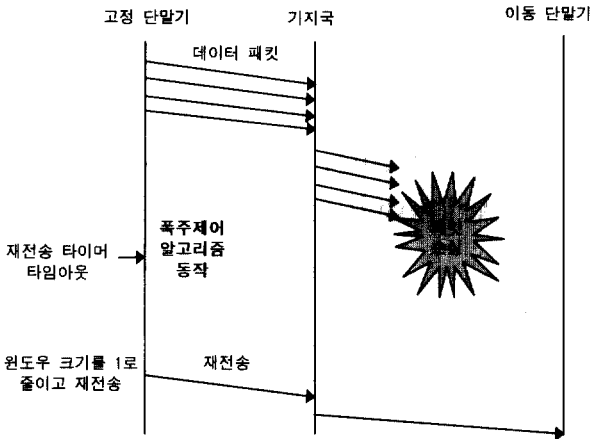
먼저 TCP 프로토콜에서 호 설정시 윈도우크기(W)는 1세그먼트로, Slow-Start Threshold(Wt)는 65,535바이트로 초기화된다. TCP 송신측에서는 데이터 패킷을 전송한 후, ACK 패킷이 타임아웃 될 때까지 도착되지 않으면 망내에 폭주가 발생된 것으로 보고, Wt는 W/2로 두고, W는 1세그먼트로 설정하여 망으로 유입되는 데이터의 양을 현저히 줄이게 된다.

또한 타임아웃이 발생되지 않고 송신측에서 새로운 ACK 패킷을 수신하게 되면 W와 Wt를 비교하여 W가 작으면 Slow Start 알고리즘이 동작되어 W 값을 1세그먼트씩 증가시키며, Wt가 크면 Congestion Avoidance 알고리즘에 따

라 W를 1/W 씩 증가시킨다.

한편, 송신측에서 3개의 동일한 ACK 패킷을 연속적으로 수신하면 수신측에서 해당 데이터 패킷만을 수신하지 못하여 대기하고 있는 것으로 간주하고, 송신측은 Fast Retransmit 알고리즘에 따라 해당 데이터 패킷부터 즉시 재전송하고 Wt는 W/2로 두고 W는 Wt로 감소시킨다. 그리고 재전송 이후 새로운 ACK 패킷이 수신되면 Congestion Avoidance 알고리즘이 동작된다.

위에서 설명한 TCP 프로토콜의 폭주제어 알고리즘에 따라 무선 인터넷 서비스를 제공하는 경우 (그림 1)과 같은 예기치 못한 상황이 자주 발생될 수 있다.



(그림 1) 무선통신 환경에서 TCP의 동작

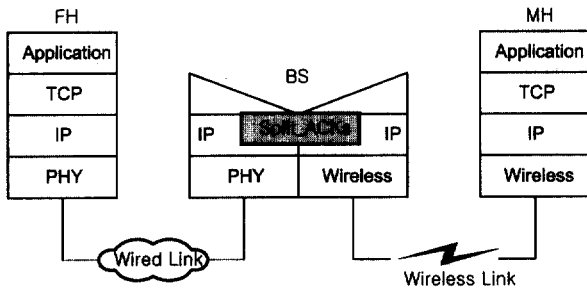
(그림 1)에서 보는 바와 같이 고정단말기는 현재의 윈도우크기 만큼 데이터 패킷을 기지국을 통해 이동단말기로 전송하였다. 그러나 기지국과 이동단말기간의 무선링크에서 패킷 손실이 발생되어 일정시간 동안 ACK 패킷이 도착되지 않으면 고정단말기의 재전송 타이머는 타임아웃이 된다. 이에 따라 TCP 프로토콜의 폭주제어 알고리즘이 동작되고, 재전송과 함께 고정단말기의 윈도우 크기는 1세그먼트로 줄게 되어 TCP 프로토콜의 성능이 급격히 저하된다. 즉, 실제 망내에 폭주가 발생되지 않았지만 무선링크의 높은 손실율로 인하여 고정단말기는 폭주가 발생된 것으로 오인하고 폭주제어 알고리즘을 동작시키게 되는 것이다. 이러한 TCP 프로토콜의 성능저하 문제를 해결하기 위해 많은 연구가 이루어졌으며 3장에서는 이들 알고리즘의 동작원리와 장단점을 기술하였다.

3. 기존의 연구

유무선 통합통신 환경에서 TCP 프로토콜의 성능저하 문제를 해결하기 위한 기존의 연구결과들은 크게 End-to-end 기법, Split-connection 기법, 및 Link Layer 기법 등의 3 종류로 분류된다.

4. 제안된 기법 - SPACK

본 논문에서 제안하는 기법은 FH(Fixed Host)와 MH(Mobile Host)에서는 TCP 프로토콜을 수정없이 그대로 사용하고, 대신 BS(Base Station)에 Split ACKs 모듈을 추가하여 TCP 프로토콜의 성능향상을 도모하였다. 먼저 Split ACKs (SPACK) 기법에서 사용되는 프로토콜 구조는 (그림 4)와 같다.



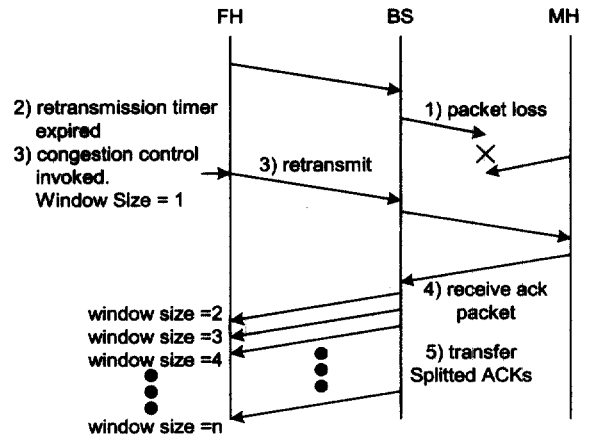
(그림 4) SPACK 기법의 프로토콜 구조

(그림 4)에 제시된 바와 같이 FH와 MH는 기존의 TCP/IP 프로토콜 구조를 그대로 사용하지만 BS에는 고유의 IP 라우팅 기능과 더불어 SPACK 모듈이 추가된다. 이러한 프로토콜 구조에서 FH와 MH간의 데이터 송수신을 위해 종단간 호선편이 이루어지고, 평상시 BS는 FH나 MH에서 보내 온 데이터를 받아 중계하는 고유의 라우팅 기능만을 수행한다.

한편, BS와 MH간의 무선링크 상에 패킷 손실이 발생되면 FH에서는 마치 폭주현상이 발생된 것으로 오인하게 되며 윈도우 크기(W)를 1세그먼트나 W/2로 줄이고 TCP 프로토콜의 폭주제어 알고리즘을 동작시켜 FH의 패킷 전송 속도를 낮추게 된다. 그 이후 줄어든 윈도우 크기는 새로운 ACK가 FH에 도착되면 Slow Start와 Congestion Avoidance 알고리즘에 따라 증가된다.

본 논문에서는 TCP 프로토콜의 폭주제어 알고리즘은 수신된 ACK (Acknowledgement)의 개수에 따라 윈도우 크기를 증가시킨다는 점에 착안하여, BS가 MH로부터 재전송된 패킷에 대한 ACK를 수신하면 이를 여러 개로 나누어 FH에 전달하여 FH의 윈도우 크기를 빠르게 복구시키는 SPACK 기법을 고안하였다. SPACK 알고리즘의 동작 예는 (그림 5)와 같다.

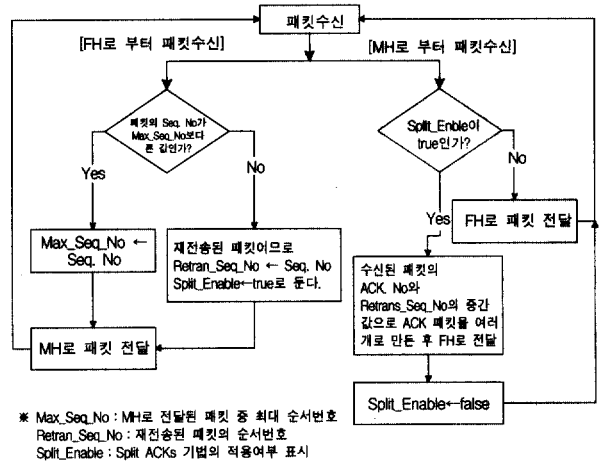
(그림 5)에서 보는 바와 같이 FH에서 전송된 데이터 패킷이 1) BS와 MH사이에서 손실되면 2) 얼마 후 FH의 재전송 타이머가 타임 아웃된다. 이에 따라 3) 폭주제어 알고리즘이 호출되고 FH의 TCP 프로토콜은 윈도우 크기를 1세그먼트로 줄인 후 재전송을 수행한다. 그리고 MH에서 데이터 패킷을 제대로 수신하게 되면 4) ACK 패킷을 FH로 전달한다. 이때 5) BS에서는 ACK 패킷을 그대로 FH에게 전달



(그림 5) SPACK 알고리즘의 동작 예

하지 않고 여러 개로 쪼개어 보내줌으로써 FH의 윈도우 크기는 빠르게 증가된다. 다음으로 (그림 6)은 SPACK 기법의 동작과정을 흐름도(Flowchart)로 보여주고 있다.

SPACK 기법은 (그림 6)에서 보는 바와 같이 BS에서는 FH로부터 패킷을 수신하면 기존의 전송된 패킷의 순서번호와 비교하여 재전송되는 패킷인지 아닌지를 판단한다. 만약 재전송되는 패킷이 아닌 경우에는 MH로 그대로 전달하지만, 재전송되는 패킷인 경우에는 패킷의 순서번호를 변수 Retrans_Seq_No에 저장하고 SPACK 기법을 활성화시킨다. 이후에 MH로부터 패킷이 수신되면 수신된 패킷내의 ACK 번호와 Retrans_Seq_No 사이의 값을 ACK 번호로 하는 ACK 패킷을 여러 개 만들어 FH로 전달하면, FH에서는 도착된 패킷의 개수에 따라 윈도우의 크기를 증가시키게 된다.

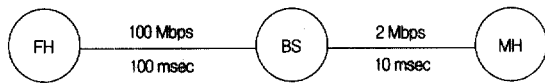


(그림 6) SPACK 알고리즘 흐름도

위에서 설명한 바와 같이 SPACK 기법은 FH와 MH의 TCP 프로토콜을 수정 없이 그대로 사용할 수 있을 뿐만 아니라, BS의 복잡도도 기존의 다른 기법에 비하여 낮은 편이다. 또한 TCP의 End-to-end Semantics를 그대로 유지하며 재전송이 중복되는 경우도 발생되지 않는다.

5. 성능분석

이 절에서는 제안한 SPACK 기법의 성능을 컴퓨터 시뮬레이션을 통하여 기존의 TCP와 비교 분석하였다. 시뮬레이션은 ns-simulator [12]를 사용하였으며 시뮬레이션 모델은 (그림 7)과 같다.



(그림 7) 시뮬레이션 모델

(그림 7)에서 보는 바와 같이 유무선 복합망의 특성을 고려하여 시뮬레이션 모델은 FH, BS 및 MH 등 총 3개의 노드로 구성하였으며, FH와 BS간의 링크는 100Mbps의 대역폭과 100msec의 전송지연 시간을 가정하였다. 또한 BS와 MH간은 2Mbps의 대역폭과 10msec의 전송지연을 가지도록 하였으며, 이 구간에서만 패킷손실이 발생하는 것으로 가정하였다.

시뮬레이션 시나리오는 먼저 FH에서 MH로 10KB~100KB 크기의 데이터를 전송하도록 하고, MSS(Maximum Segment Size)도 250B에서 1500B까지의 값으로 변화를 주었다. 그리고 BS와 MH 구간에서 패킷손실율을 1%에서 10%까지를 적용하여 시뮬레이션을 수행하였다. 또한 SPACK 기법의 주요인자인 응답패킷의 개수도 여러 값을 주어 성능 분석에 활용하였다.

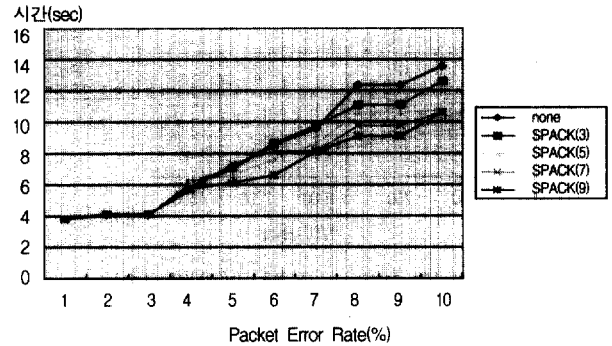
앞서 설명된 기존의 연구결과들은 TCP 프로토콜의 수정, End-to-end Semantics의 파괴, 재전송 중복 등의 문제점을 가지므로, 이러한 문제점이 없는 SPACK 기법을 적용한 TCP 프로토콜과 SPACK을 적용하지 않은 기존의 TCP 프로토콜만을 비교 분석하였다.

시뮬레이션 결과를 보면 먼저 (그림 8)과 (그림 9)는 각각 전송데이터의 크기가 100KB인 경우에 MSS가 1000B, 500B인 환경에서 패킷 손실율과 전송시간과의 관계를 보여주고 있다. 이 두 개의 그래프에서 패킷손실율이 증가할수록 기존의 TCP보다 SPACK을 적용한 기법의 전송시간이 짧음을 알 수 있다.

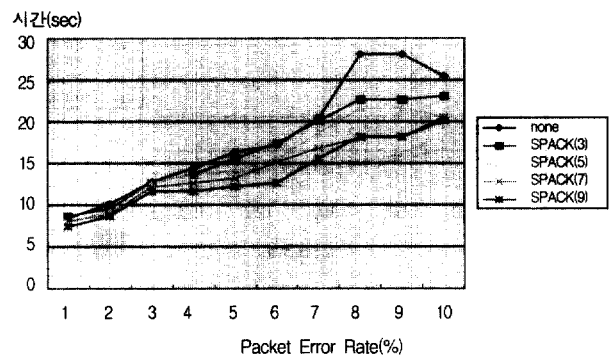
(그림 10), (그림 11)은 각각 MSS가 500B인 경우 패킷손실율이 5%, 10%인 환경에서 전송되는 데이터의 크기를 10KB에서 100KB로 증가시키면서 전송시간을 측정한 결과이다. 이 그래프에서도 기존의 TCP 프로토콜에 비하여 SPACK 기법을 적용한 기법이 더 적은 전송시간이 소요됨을 보여주고 있다.

다음으로 (그림 12), (그림 13)은 패킷손실율이 5%인 환경에서 100KB와 500KB의 데이터를 전송하는 경우 MSS를 250B에서 1500B로 증가시켜 가면서 전송시간을 측정한 결과이다. 이 그래프에서도 SPACK 기법을 이용하는 경우 기

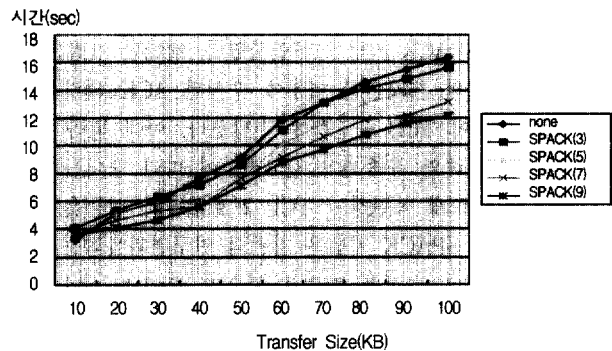
존의 TCP 프로토콜을 그대로 사용하는 것보다 전송시간이 적음을 확인할 수 있다.



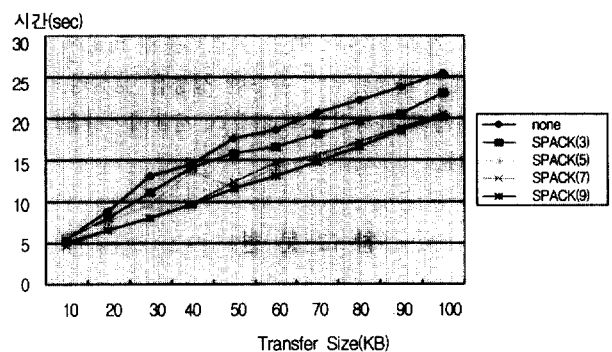
(그림 8) 패킷손실율 vs. 전송시간(MSS=1000B)



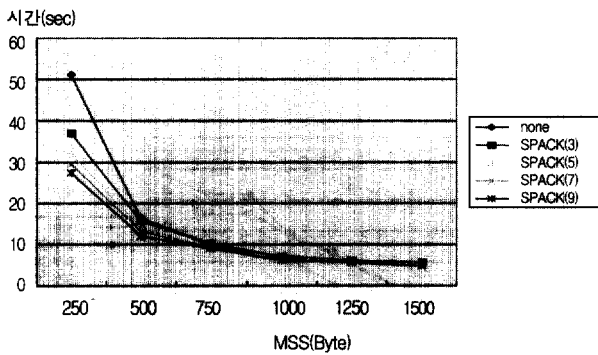
(그림 9) 패킷손실율 vs. 전송시간(MSS=500B)



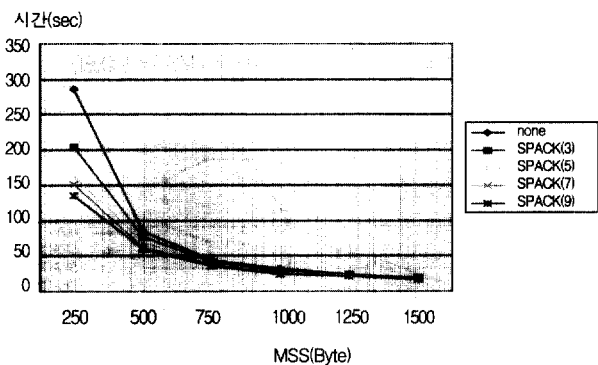
(그림 10) 데이터크기 vs. 전송시간(5%손실)



(그림 11) 데이터크기 vs. 전송시간(10%손실)



(그림 12) MSS vs. 전송시간(100KB전송)



(그림 13) MSS vs. 전송시간(500KB전송)

6. 결 론

본 논문에서는 무선환경에서 TCP 프로토콜의 성능을 개선하기 위한 SPACK 기법을 제안하였다. 제안된 기법은 BS에서 무선링크의 패킷손실 이후에 수신된 ACK 패킷을 여러 개로 나누어 TCP 송신측에 전달한다. 이에 따라 TCP 프로토콜은 기존의 폭주제어 알고리즘에 따라 줄어든 윈도우 크기를 빠르게 복원하여 TCP 프로토콜의 성능을 향상시킬 수 있다. SPACK 기법은 기존의 연구결과들과는 달리 TCP 프로토콜을 수정할 필요가 없으며, End-to-end TCP Semantics도 유지된다. 또한 BS의 복잡도도 낮을 뿐만 아니라 재전송이 중복되는 현상도 발생되지 않는 장점을 가지고 있다.

제안된 기법의 성능을 검증하기 위하여 시뮬레이션을 수행하였으며, 그 결과 기존의 TCP 프로토콜에 비하여 예러가 많이 발생하는 환경에서도 우수한 성능을 보였다. 앞으로는 인터넷 환경과 유사하도록 컴퓨터 시뮬레이션을 보다 강화할 예정이다.

참 고 문 헌

[1] 한국인터넷정보센터(KRNIC), <http://www.nic.or.kr/>,

[2] Kevin Thompson, Gregory J. Miller, Rick Wilder, "Wide-Area Internet Traffic Patterns and Characteristics," IEEE Network, pp10-23, Nov. 1997.
 [3] "무선 인터넷 가입 2,150만명 넘어서", 한국인터넷정보센터(KRNIC), <http://www.nic.or.kr/>, Sept. 2001.
 [4] W. Richard Stevens, *TCP/IP Illustrated*, Vol.1, Addison Wesley, 1994.
 [5] 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.
 [6] Daichi Funato, Kinuko Yasuda and Hideyuki Tokuda, "TCP-R: TCP Mobility Support for Continuous Operation," Proceedings of International Conference on Network Protocols, pp.229-236, Oct. 1997.
 [7] A. Bakre and B. R. Badrinath, "I-TCP: Indirect TCP for Mobile Hosts," Proceedings of 15th International Conference on Distributed Computing Systems, Vancouver, Canada, pp.136-143, May, 1995.
 [8] K. Brown and Suresh Singh, "M-TCP: TCP for Mobile Cellular Networks," ACM SIGCOMM, Computer Communication Review, pp.19-43, July, 1997.
 [9] Kuang-Yeh Wang and Satish K. Tripathi, "Mobile-End Transport Protocol an Alternative to TCP/IP over Wireless Links," Proceedings of INFOCOM'98, Vol.3, pp.1046-1053, April, 1998.
 [10] Elan Amir, Hari Balakrishnan, Srinivasan and Randy H. Katz, "Efficient TCP over Networks with Wireless Links," Proceedings of 5th Workshop on Hot Topics in Operating Systems, pp.35-40, May, 1995.
 [11] E. Ayanoglu, S. Paul, T.F. Laporta, K. K. Sabnani, and R. D. Gitlin, "AIRMAIL: A Link-Layer Protocol for Wireless Networks," ACM/Baltzer Wireless Networks Journal, pp. 47-60, Feb. 1995.
 [12] "UCB/LBNL/VINT Network Simulator-ns(version2)," <http://www-mash.cs.berkeley.edu/ns.>



진 교 흥

e-mail : khjin@dongeui.ac.kr

1991년 부산대학교 컴퓨터공학과(공학석사)
 1993년 부산대학교 컴퓨터공학과(공학석사)
 1997년 부산대학교 컴퓨터공학과(공학박사)
 1997년~2000년 국방과학연구소 선임연구원
 2000년~현재 동의대학교 멀티미디어공학과
 전임강사

관심분야 : TCP/IP, LAN, 초고속통신망 등