기지국에서의 지역 ACK 제어를 이용한 효율적인 무선 TCP의 성능 분석

*박진완, *윤종호
*한국항공대학교 항공통신정보공학과

Performance analysis of an efficient TCP with Local ACK control at an Access Point over wireless links

*Jin-Woan Park. *Chong-Ho Yoon
*Dept. of Telecommunication & Information Engineering. Hankuk Aviation University.

요 약

본 논문에서는 유무선 통합망에서 ACK 제어를 통해 이동 호스트의 핸드오프로 인한 TCP의 성능 저하를 방지하는 방안을 제안한다. 이동 호스트의 핸드오프 발생 시, 기지국이 ACK 패킷 내의 광고 윈도우의 값을 0으로 설정, 고정 호스트에게로 전송하여 고정 호스트를 persist 모드로 동작하게 하여 불필요한 혼잡제어를 회피하고 핸드오프가 종료된 이후에 기존의 혼잡 윈도우와 재전송 타이머를 유지하여 가용 대역폭을 효율적으로 사용할 수 있다. 모든 기능을 기지국으로 한정하므로 종단의 TCP변경이 필요 없는 장점이 있으며 실험을 통해 동작 및 성능을 검증하였다.

Ⅰ. 서론

무선 통신 기술의 발달과 사용자의 이동성에 대한 요구증대로 인해 기존의 인터넷 환경은 유.무선이 통합되는 단일 망으로 변화하고 있고, 현재 유선 네트웍 환경에서 ftp, telnet, WWW와 같은 어플리케이션들이 모두 TCP에 기반을 두고 있어서 TCP의 성능은 무선 네트웍의 이용에 있어서도 중요한 요소가 될 것이다.
현재의 TCP/IP 프로토콜은 유선 링크로 구성된 통신망에서 부하가 동적으로 변하는 상황에서 발생하는 오류에 잘 적응할 수 있는 프로토콜로 전송 오류율이 낮은 통신 매체를 근간으로 하였기 때문에 데이터 전송 시에 발생하는 모든 오류는 통신망의 과부하, 즉 폭주로 인한 데이터 전달 경로의 중간 라우터에서 패킷을 폐기시키기 때문에 발생하는 것으로 가정한다. 유선망에서 TCP는 이러한 오류 발생에 효과적으로 대처하기 위해 통신망의 부하에 따

라 전송 속도를 조절하는 폭주 제어 방식을 이용한다.[1].
그러나 유선망과 달리 무선망을 잡음, 페이딩, 간섭 등에 의한 높은 비트 오류율과 이동 호스트의 이동에 의한 일시적인 연결 두절 및 핸드오프 등 여러 다른 요인에 의해 패킷 손실이 발생할 수 있다. 그러므로 만약 유무선 복합망에서 패킷 손실의 발생 시 기존의 TCP를 적용하면 모든 패킷 손실을 폭주에 의해 발생했다고 고려하여 폭주 제어를 수행한다.
이 과정에서 무선망 환경의 특징에 의해 발생한 패킷 손실을 폭주에 의해 발생했다고 보게 되면 데이터의 전송에 있어서 신뢰성과 함께 효율성을 보장하지 못해 유선 구간의 대역폭 이용 면에서 불필요한 감소를 초래하여 저리율의 저하 및 지연을 증가시켜 전체망의 성능을 저하시키게 된다.
무선환경 상에서 TCP에 가장 큰 영향을 끼치는 현상들은 핸드오프등에 의해 발생하는 집중적인에러로 인한 급격한 윈도우 크기의 감소, 이에 이은 TCP의 슬로우 스타트의 영향으로 인한 채널 용량의 낭비를 들 수 있다. 에러의 발생 자체는 환경 고유의 영향으로 막을 수 없으나, Snoop 등의 프로토콜로 TCP에 끼치는 영향을 상당히 줄일 수 있다[2].
본 논문에서는 Snoop와 같은 프로토콜들이 무선 구간의 간헐적인 패킷 손실에 잘 대처한다고 보고 무선망에서 발생되는 핸드오프에 대해서만 그 중점을 두었다. 제안된 방법에서는, 핸드오프 발생 시 기지국에서 ACK 패킷을 제어하여 TCP 송신측을 Persist모드로 만들어 TCP 송신축의 재전송 타이머와 혼잡 윈도우를 이전 상태로 유지하여 무선망에서 발생되는 핸드오프로 인한 TCP 성능 저하를 방지한다.
본 논문의 구성은 다음과 같다. Ⅱ장에서는 무선환경에서의 TCP성능 향상을 위한 대표적인 연구와 그 문제점을 설명한다. Ⅲ장에서는 무선망에서의 일반적인 핸드오프 수행절차와 이로 인한 문제점을 살펴보고 Ⅳ장에서 기지국에서의 ACK제어를 이용한 개선방안을 설명한다. Ⅴ장에서 모의 실험을 통해 제안한 방식의 성능을 분석하고 마지막으로 Ⅵ장에서는 결론을 내린다.

Ⅱ. 기존의 무선 TCP 프로토콜

무선환경 하에서 TCP성능 향상을 위한 대표적인 프로토콜들은 다음과 같다.
Fast Retransmission 프로토콜은 핸드오프의 완료 후 마지막으로 수신한 패킷에 대해 즉시 3개의 중복 ACK를 전송하거나 특정의 핸드오프의 완료 메시지를 전송하여 송신측으로 하여금 타임아웃 발생이전에 fast retransmission이 발생되도록 하여 핸드오프로 인한 지연을 줄이는 방법이다. 그러나 핸드오프 이후에 이전의 대역폭에 도달하기 위해서는 몇 번의 RTT가 경과해야 하며 인위적인 fast retransmission의 수행을 위해 종단 호스트의 기존 TCP 변경이 필요하다[3].
I-TCP는 대단(End to End) TCP연결을 고정 호스트와 기지국사이의 유선연결, 기지국과 이동 호스트사이의 무선연결로 나누어서 각각의 연결에 적합한 정책을 적용 함으로써 전송 성능의 향상을 이루고자 하는 방법이다. 핸드오프 발생 시 이전 기지국이 유, 무선구간의 두 개의 연결상태정보와 ACK 받지 못한 패킷들을 새로운 기지국으로 전달한다. 그러나 상태정보의 전달과 재설정에 의한 핸드오프 지연이 크고 종단간 대칭성이 위반된다[4].
Snoop는 이동 호스트와 고정 호스트 사이에서 패킷 송수신을 책임지는 기지국에서 Snoop Agent를 통해 패킷 손실 시 지역 재전송을 통해 전송성능의 향상을 이루어내는 방법이다. 핸드오프를 위해 이동 호스트가 현재 속해 있는 셀과 그 주변 셀을 하나의 멀티캐스트 그룹으로 구성하고 송신호스트는 보내려는 패킷을 멀티캐스트 그룹에 전송하는 Multicast & Intelligent buffering 방식을 취하는데 이를 위해 기지국은 필요이상의 많은 캐쉬를 유지해야 한다[5].
M-TCP는 TCP와 같은 연결 분리 방식으로 이동 호스트로부터 일정시간 일정 시간 ACK를 받지 못할 경우 송신측으로 윈도우 크기가 0인 ACK를 전송하여 송신측을 Persist모드로 전환하게 한 방법이다. 그러나 타이머에 의존에 의해 신속적으로 대처하지 못할 수 있으며 이동국은 슬로우 스타트를 적용하지 않아 대역폭 관리를 위한 추가적인 요소가 필요하다.[6]

Ⅲ. 무선망에서의 핸드오프 수행 절차

무선망에서의 핸드오프 수행 절차는 <그림 1>과 같다. 기지국은 주기적으로 beacon 메시지를 보내고 이 beacon 메시지를 받는 이동 호스트는 자신이 어느 셀에 있는 지 알게 된다. 이 때, 자신이 받고 있는 beacon 메시지 보다 더 강한 신호의 beacon 메시지를 수신한 경우 핸드오프가 일어났음을 알 수 있다. 이 때 이동 호스트는 greeting 메시지를 보냄으로써 새로운 기지국에게 자신의 존재를 알려주게 되고, 새로운 기지국은 이전의 기지국에게 핸드오프 사실(Inform)을 알려준 후 라우팅 정보 등의 해당 정보들(Adjust)을 가져오게 된다[7][8].

<그림 1> 핸드오프 수행절차.

이렇게 핸드오프가 발생하였을 경우, 기존의 기지국은 새로운 기지국으로부터 Inform 메시지를 받기 전까지 이동 호스트가 자신의 셀 영역을 떠났다는 것을 알지 못하며 새로운 기지국도 이동 호스트로부터 Greeting 메시지를 받기 전까지는 기존의 기지국에 핸드오프가 발생했다는 Inform 메시지를 보내지 못한다. 따라서 기존의 기지국은 이동 호스트의 이동을 알기 전까지는 이동 호스트에 계속해서 패킷을 전송하게 되며 이러한 패킷은 기존의 기지국이 이동 호스트에 도달하지 않기 때문에 손실된다. 이러한 핸드오프 기간 중 발생되는 패킷의 손실에 대해 기존 TCP는 이를 혼잡상황으로 간주하므로 핸드오프 과정의 시간 지연에 따른 송신기의 재전송 타이머의 타임아웃을 발생시켜 슬로우 스타트 단계를 수행하여 전송 윈도우 크기를 줄여 망 혼잡을 회피하려 하게 된다. 무선망에서 이동국의 이동성으로 인한 핸드오프에 의해 이루어지는 이러한 동작은 핸드오프가 종료된 이후에도 이전의 전송율로 회복하는 데 상당한 시간을 필요로 하므로 유선 구간의 대역폭 이용 면에서 불필요한 감소를 초래하며 처리율의 저하 및 지연을 증가시켜 전체 망의 성능을 저하시키게 된다.

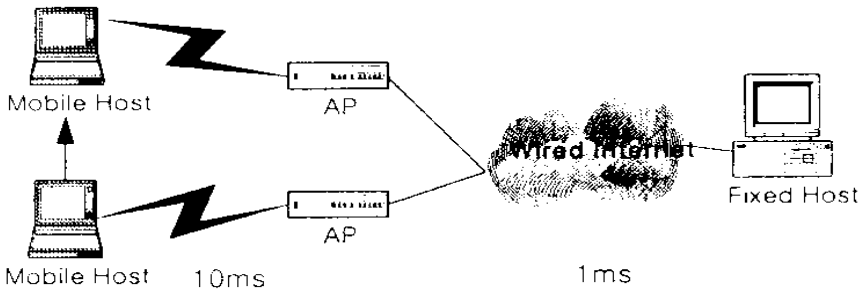
Ⅳ. ACK제어를 이용한 Mobile TCP

핸드오프 중이라도 송신측으로 하여금 재전송 타이머가 만기 되지 않게 한다면 이로 인한 불필요한 혼잡제어의 수행을 억제할 수 있고 유선 구간의 대역폭 낭비를 줄여 수 있다. 이러한 무선망에서의 특성으로 인한 불필요한 혼잡제어의 회피를 위해 유선망과 무선망이 혼재하는 환경에서, 유선망의 호스트들이 이동 호스트의 존재를 인지하여 유.무선망을 하나의 통합된 단일망이 되도록 프로토콜을 수정하는 것이 가장 바람직하지만, 이동 호스트를 지원하기 위해 인터넷의 전체 호스트를 수정하는 것은 결코 좋은 방법일 수 없다. 그러므로, 망의 성능 저하를 막기위해서는 무선구간에서 발생한 손실을 무선구간 자체 내에서 해결하여 유선망의 호스트가 이동 호스트의 존재를 인터넷상의 같은 호스트로 인식하게 해야 한다. 따라서 유 무선망의접속 지점인 기지국에서 이러한 기능을 수행하는 것이 타당하다.
본 논문에서는 RFC1122에서 TCP의 필수 사항으로 규정한 Persist모드를 이용하여 핸드오프 시 재전송 타이머의 만기를 억제 한다. 먼저 기지국은 고정 호스트로부터 전달된 패킷에 대해 이동 호스트로부터 ACK패킷이 도달할 때 까지 이를 저장하고 이동 호스트가 고정 호스트에게 보내는 해당 ACK 패킷을 수신하면 이를 삭제하고 이동 호스트로부터의 재전송 요구에 대해 ACK받지 못했던 패킷이면 이를 지역 재전송한다. 그리고 ACK패킷은 전송 후 새로운 ACK패킷이 도달하기 전까지 핸드오프를 위해 저장한다. 만약 이동 호스트가 이전

기지국보다 강력한 Beacon을 수신하면 핸드오프 과정이 시작되고 Greeting 메시지를 수신한 새로운 기지국은 이전 기지국에게 핸드오프의 발생을 알리는 Inform 메시지를 전송한다. 이전의 기지국이 이동 호스트의 이동을 최초로 알게 되는 시점이 이 지점이며, 이 때 이전 기지국은 보관하고 있던 마지막으로 ACK받은 ACK패킷 내의 광고 위도우의 값을 "0"으로 설정하고 고정 호스트에게로 전송하여 고정 호스트를 persist mode로 전환하게 한다. 그리고 Adjust 메시지를 이전 기지국에게 전송함과 동시에 이전 기지국에 ACK받지 못하여 저장되어 있는 패킷들을 새로운 기지국으로 전송하여 나머지 핸드오프 과정을 수행한다. 고정 호스트는 ACK 패킷 내의 0인 광고 윈도우의 값에 의해 상대 호스트가 패킷 수신을 할 수 없다고 판단하고 양의 광고 윈도우 값을 수신할 때까지 재전송 타이머 및 혼잡 윈도우 크기를 이전 상태로 유지시킨 상태에서 패킷의 전송을 중지하게 된다. 핸드오프의 종료 후 이전 기지국으로부터 전달 받은 ACK 받지 못했던 패킷을 새로운 기지국이 호스트로 전송하므로 이동 호스트는 양의 광고 윈도우로 ACK하게 되고 고정 호스트는 양의 광고 윈도우의 수신에 의해 핸드오프 이전에 사용하던 윈도우 크기와 임계치(ssthresh). 재전송 타이머 값을 그대로 유지함으로써 묵시적으로 핸드오프의 발생을 고정 호스트로부터 숨겨 TCP 성능을 전과 같이 유지 할 수 있다.

Ⅴ. 모의 실험 및 성능 분석

제안된 방법의 성능을 확인하기 위해 핸드오프가 일어날 때 TCP 연결의 성능을 모의 실험하였다. 모의 실험은 Linux 6.0 환경에서 객체지향 언어인 Simula을 사용하였으며 객체는 TCP 송, 수신 호스트, 패킷 손실 시 지역재전송과 핸드오프 발생 시 ACK를 제어를 수행하는 기지국, 링크로 구성된다.

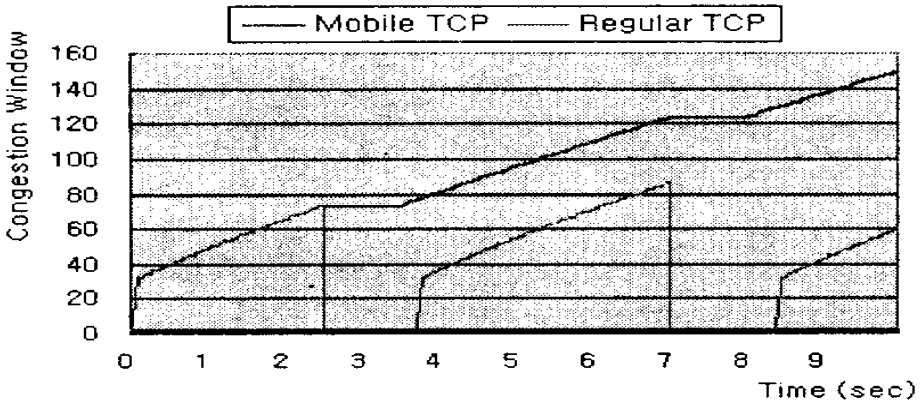


<그림 2> 실험환경.

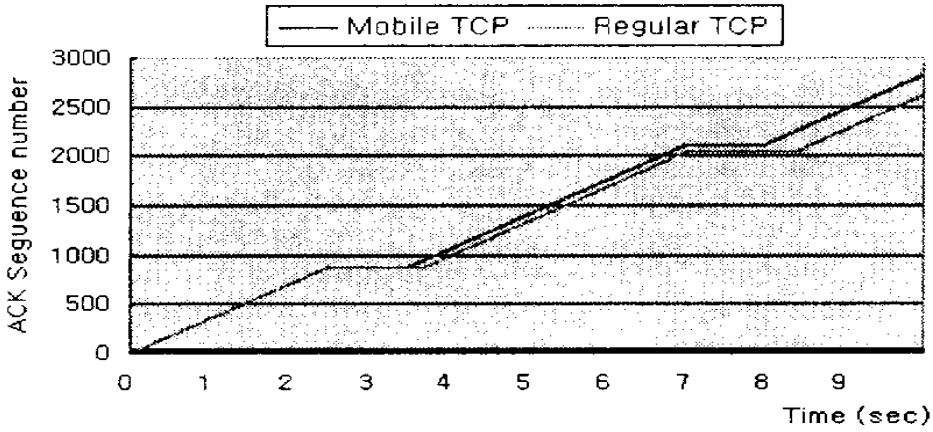
<그림 2>의 망 배치도를 바탕으로 유,무선 구간에 걸친 TCP 연결의 성능을 분석하였다. 10초의 모의실험 동안에 2.5초와 7초의 시점에서 1초 기간의 핸드오프가 2번 발생하게 하였다. 유,무선 구간에서의 전송속도는 1.5Mbps, 지연은 각각 1ms, 10ms로 설정하였다. 패킷의 크기는 512bytes, 최대 윈도우 크기는 32 패킷으로 설정하였으며 고정 호스트에서 이동 호스트로의 벌크 데이터 전송만을 고려한다.

그러므로, TCP 송신자는 고정 호스트에 내장되어 있고, TCP 수신자는 이동 호스트에 내장되어 있다. 그리고 유선망에서는 혼잡이 없다고 가정하고 핸드오프가 수행되는 동안 패킷은 이전 기지국에서 저장되었다가 핸드오프가 종료된 이후에 이동 호스트로 전송되도록 하였다.

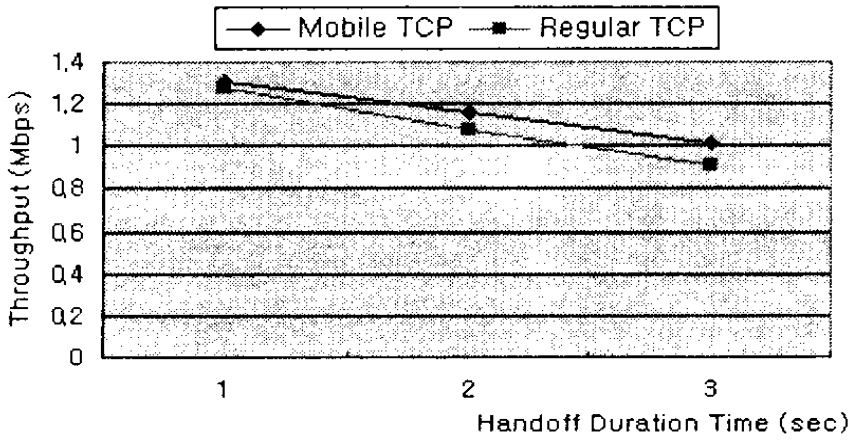
<그림 3>과 <그림 4>는 기존의 TCP와 제안된 방식의 윈도우 크기와 고정 호스트가 기지국을 통해 전송한 패킷에 대해 이동 호스트가 고정 호스트로 전송한 ACK의 수를 비교한 것으로 핸드오프가 발생될 때마다 기존 TCP 방식은 패킷 손실로 인한 재전송 타임아웃에 의해 슬로우 스타트를 수행하여 초당 ACK의 갯수도 느리게 증가하는데 반해 제안된 방식에서의 혼잡 윈도우 크기는 Persist 모드의 전환으로 인해 핸드오프로 인한 영향을 거의 받지 않으며, 핸드오프의 발생 시에만 전송이 중단되고 핸드오프가 완료된 이후에는 고정 호스트에서 이전의 혼잡 윈도우 크기를 그대로 사용함으로써 핸드오프 이전에 사용하던 대역폭으로 데이터를 전송하고 있음을 알 수 있다.



<그림 3> 혼잡 윈도우의 비교.



<그림 4> ACK Seguence number의 비교.



<그림 5> 핸드오프 기간에 따른 초당 처리율의 비교.

<그림 5> 는 핸드오프 기간과 처리율의 관계로 기존의 TCP 방식은 핸드오프 기간과 송신기의 재전송 타이머의 값에 따라서 성능의 차이가 크게 변화하는 반면, 제안하는 방식은 핸드오프 기간에 따라 처리율이 일정하게 감소한다. 이것은 핸드오프동안 기존의 기지국에 핸드오프 이전에 버퍼링된 패킷이 새로운 기지국을 통해서 이동 호스트로 전송되기 때문이다. 일반적으로 핸드오프 시간이 길어질수록 전체적인 Throughput이 감소함을 알 수 있다.

Ⅵ. 결론

본 논문에서는 유무선 통합망에서 ACK제어를 통해 이동국의 핸드오프과정에서 핸드오프가 완성된 이후에도 전에 사용하던 대역폭을 그대로 사용하여 망의 성능 저하를 최소한으로 줄이는 방안을 제안하였다. 이동국의 핸드오프 발생시, 기지국이 ACK 패킷 내의 광고 윈도우의 값을 0으로 설정, 고정 호스트에게로 전송하여 persist mode로 동작하게 하여 불필요한 혼잡제어를 회피하고 핸드오프가 종료된 이후에 기존의 혼잡제어 윈도우의 크기와 재전송 타이머를 유지하여 가용 대역폭을 효율적으로 사용할 수 있으며 모든 기능을 기지국으로 한정하여 종단의 TCP 변경 없이 기존 TCP 성능을 유지함을 실험은 통해 검증하였다.

참 고 문 헌

[1] Van Jacobson, Michael J. Karels, "Congestion Avoidance and Control,: SIGCOMM 88.

[2] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A Comparison

of Mechanisms For Improving TCP Performance Over Wireless Links." *IEEE/ACM Transactions on networking*, vol.5, no.6, 1997.
[3] R. Caceres and L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments," *IEEE journal on Selected Areas in Communications*, 1995.
[4] A. Bakre and B. R. Badrinath, "Handoff and system support for indirect TCP/IP," *Proc, Second Usenix Symp, on Mobile and Location-Independent Computing*, 1995.
[5] H. Balakrishnan, S. Seshan, and R.H. Katz, "Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks." *ACM Wireless Networks*, 1(4), 1995.
[6] Kevin Brown, Suresh Singh, “M-TCP: TCP for Mobile Cellar Networks," *Computer Communication Review*, V.27 N.5, 1997.
[7] Chander Dhawan, "Mobile Computing," MacGraw-Hill, 1997.
[8] 이상연, 임행빈, 정충교, “핸드오프로 인한 유무선 혼합 TCP 연결의 성능 저하 방지 방안,” 한국 통신 학회 논문집, 제18권 제1호, 1998.