IEEE 802.16 시스템에서 ARQ 타이머가 TCP 성능에 미치는 영향*

이지훈O, 박소영, 박철현, 권태경, 최양희
서울대학교
{jhlee O, sypark, chpark}@mmlab.snu.ac.kr, {tkkwon, yhchoi}@snu.ac.kr

The Effects of ARQ Timers on TCP Performance in IEEE 802.16 Systems*

Ji Hoon Lee O, Soyoung Park, Chulhyun Park, Taekyung Kwon, and Yanghee Choi
Seoul National University

요 약

본 논문에서는 IEEE 802.16 시스템에서 ARQ(Automatic Repeat reQuest)의 동작이 TCP 성능에 끼치는 영향을 분석 한다. IEEE 802.16 표준에서는 MAC 계층에서 패킷 전송의 성공 혹은 실패를 확인하여, 실패 시 패킷을 재전송하는 ARQ 기법을 포함하고 있다. 이러한 무선 링크상의 재전송은 TCP와 맞물려 동작할 때, 적절한 ARQ 재전송 타임아웃(timeout)이 보장되지 않으면 심각한 TCP 성능 저하를 초래할 수 있다. 이에 대하여 IEEE 802.16 시스템의 ARQ는 총 4가지의 타이머(timer)들이 존재하지만 그 구체적인 값들은 표준에서 다루고 있지 않다. 본 연구는 이러한 ARQ 타이머들의 값에 따라 TCP 성능이 어떻게 변하는지를 시뮬레이션을 통해 분석하고 적절한 값을 제시한다.

1. 서 론

전송계층 프로토콜 중 신뢰성 있는 연결 지향성 수송 프로토콜인 TCP는 자체적인 흐름 제어 알고리즘과 혼잡 제어 알고리즘을 포함하고 있다. 그러나 이 알고리즘은 링크 계층 오류가 거의 없는 기존 유선 망에서의 동작을 위주로 개발되어 상대적으로 링크 계층 오류가 큰 무선 망에서는 잘못된 혼잡 제어 알고리즘 구동으로 인한 성능 저하를 보이게 된다. 무선 MAC 계층의 ARQ (Automatic Repeat request) 기법은 TCP가 무선 링크상의 오류를 알지 못하도록 무선 링크 구간에서의 오류를 ARQ 블록 단위 재전송을 통해 복구한다. 이는 TCP의 혼잡 제어 알고리즘이 부적절히 구동되는 것을 막아 TCP의 성능을 향상시킨다.

한편, 넓은 대역폭을 요구하는 멀티미디어를 포함하여, 기존 유선 네트워크에서와 같은 고품질의 인터넷 서비스에 대한 요구가 무선 이동 통신에서도 커짐에 따라, IEEE 802.16 시스템을 기반으로 한 광대역 무선 통신망이 근래 주목을 받고 있다[1][2]. IEEE 802.16 시스템에서는 MAC 계층에서의 신뢰성 있는 전송을 보장하기 위하여 ARQ를 지원하고 있으며, 다수의 설정 가능한 타이머들을 도입하여 망이나 트래픽의 특성에 따라 적응적으로 동작시킬 수 있도록 허용하고 있다. 그러나 이렇게 높은 자유도로 인하여 타이머들을 어떻게 설정해야 할 것인가에 대한 문제가 생기고, 부적절한 값들이 선택되었을 경우, TCP와 같은 상위 계층 전송 프로토콜의 성능에 심각한 저하를 초래할 수 있다.

현재의 IEEE 802.16 시스템들은 고정 및 이동 무선 접속 환경에서 IP를 기반으로 한 인터넷 서비스가 주요한 목표이기 때문에, TCP의 성능을 고려하여 ARQ 타이머의 값들을 제시하는 것이 가장 중요하다. 본 논문에서는 ARQ 타이머들에 따라 TCP 성능이 어떻게 변하는지를 시뮬레이션을 통해 분석하고, 적절한 값을 제시한다.

본 논문의 구성은 다음과 같다. 2장 관련 연구에서는 무선망에서의 TCP 성능 향상과 관련된 기법들을 살펴보고, 3장에서는 IEEE 802.16 시스템에서의 ARQ 동작과 TCP가 연동되었을 경우의 문제점을 지적한다. 4장에서는 시뮬레이션 모델과 ARQ 구현에 대해 간략히 설명하고, 5장에서는 결과를 분석한다. 끝으로 6장에서 결론을 맺는다.

2. 관련 연구

TCP 프로토콜은 통신의 양 끝단 사이에서 일어나는 패킷 손실을 전송 중간의 오류로 인해서 발생하는 경우보다 네트워크의 혼잡(congestion)으로 인해 중간 라우터 발생하는 경우로 가정한다. 이에 따라 패킷을 전송한 뒤 일정 시간 동안 해당 패킷에 대한 응답(ACK; ACKnowledgement)이 돌아오지 않을 경우, 한번에 전송할 수 있는 패킷 윈도우(window)의 크기를 줄이는

*본 연구는 지식경제부 및 정보통신연구진흥원의 IT신성장핵심동력기술개발사업[2007-F-038-02, 미래 인터넷 핵심기술 연구]과 21세기 프론티어 연구개발사업의 일환으로 추진되고 있는 지식경제부의 유비쿼터스컴퓨팅 및 네트워크 원천기반기술 개발 사업의 08B3-B3-10M 과제로 지원된 것임. 또한 서울대학교 컴퓨터연구소로부터 연구 장비를 지원받고 공간을 제공 받았음.

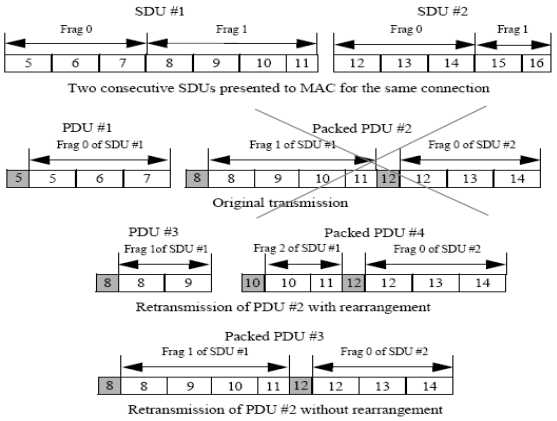

그림 1. ARQ 블록으로 구성된 IEEE 802.16 MAC PDUs

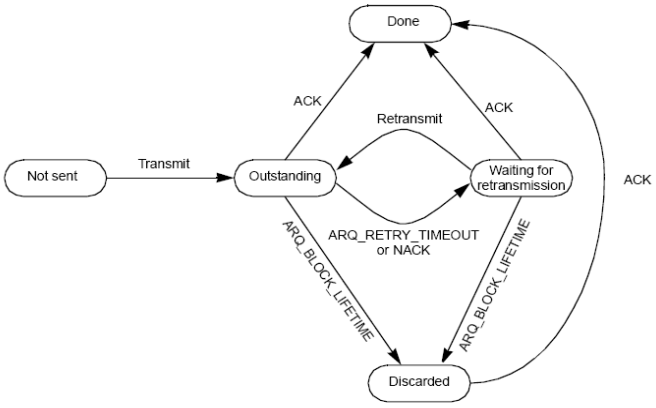

그림 2. ARQ 송신측 상태 천이도

방식으로 전송 비율을 조정한다[3]. 그러나 무선 환경에서는 네트워크의 부하 이외에도 링크 자체의 신뢰성이 떨어지기 때문에, 실제로는 네트워크에 부하가 없음에도 TCP가 전송 윈도우를 줄여서 성능을 제대로 발휘하지 못하는 단점이 존재한다.

이러한 문제점을 해결하기 위해서 무선 환경에서 TCP 프로토콜의 성능을 보완하기 위한 다양한 연구가 진행되었는데, 그 중의 하나로 TCP의 혼잡 제어 기능과 패킷 전송 기능을 분할하고, 경로상의 노드들이 프록시(proxy)로 작용하여 전송되는 패킷을 저장해 두는 기법이 있다[4]. TCP 연결은 이 프록시들 사이에서 일어나기 때문에, 무선 통신 환경으로 인한 패킷 손실이 TCP의 혼잡 제어에 영향을 미치는 범위가 줄어든다. 그러나 이러한 접근은 TCP가 갖고 있는 End-to-end 전송 형태를 해치게 된다.

한편으로, 유선망에 있는 서버로부터 전송되어 오는 패킷을 기지국의 네트워크 계층에서 검사하여, 무선 단말로부터 전송되어 오는 ACK 패킷이 중복된 ACK일 경우 기지국이 미리 임시 저장을 해 둔 패킷을 재전송하여 불필요하게 서버가 TCP의 전송률을 조정하는 일을 막도록 하는 방법도 연구된 바 있다[5].

위 연구들은 무선 통신 환경에서 TCP의 성능을 개선하기 위해서 전송 계층과 네트워크 계층에서 제시된 방법이고, 한편 전송시의 전력 조절 및 FEC와 ARQ 등을 사용하여 링크 계층에서의 패킷 손실을 보완함에 따라 TCP 프로토콜이 어떠한 영향을 받는지에 대해 살펴본 연구도 있다[6]. 본 연구에서는 IEEE 802.16 표준에서 기본적으로 제공하고 있는 링크 계층 ARQ를 적절히 사용함으로써 TCP 성능 향상을 추구하며, 유효한 ARQ 타이머 값의 범위를 제시한다.

3. IEEE 802.16 MAC 계층 ARQ 동작

IEEE 802.16에서 단말기는 최소한 1개 이상의 connection을 가지며, 각 트래픽 플로우(traffic flow)에 따라 별도의 connection을 할당 할 수도 있다. IEEE 802.16의 MAC 계층 ARQ는 각 connection 별로 동작시킬 수 있는데, 동작 여부는 각 connection을 생성할 때, 단말기와 기지국간의 capability negotiation을 통해서 정해진다. ARQ가 동작하도록 설정된 경우, IEEE 802.16 MAC 계층에서는 먼저 MAC SDU (Service Data Unit)를 ARQ 블록(block) 단위로 나눈다. 한 개 이상의 ARQ 블록들이 MAC 계층 스케줄링에 따라 모아지고, IEEE 802.16 MAC 헤더를 붙이면, 비로소 하나의 MAC PDU (Protocol Data Unit)가 된다. 그림 1은 ARQ를 사용한 경우, MAC SDU #1과 #2가 각각 PDU #1과 Packed PDU #2로 만들어진 것을 보여준다. 또, PDU #2가 전송 실패한 경우, PDU #3과 Packed PDU #4 혹은 Packed PDU #3 만으로 재전송 할 수 있음을 보여준다. 여기서 각 PDU내의 숫자들이 ARQ 블록 순서 번호(block sequence number)를 의미하고, 수신측에서는 이 번호에 대해 ACK 혹은 NACK으로서 응답한다.

그림 2에서는 송신측 ARQ의 상태 천이도를 보여준다. 앞서 설명한 바와 같이 만들어진 IEEE 802.16 PDU가 전송이 되면 (Outstanding 상태), 송신측에서는 ARQ 수신 측으로부터 응답을 기다리게 되며, NACK (Negative ACK)이 오거나 혹은 ARQ_RETRY_TIMEOUT이 지난 경우(Waiting for retransmission 상태)에 재전송을 하고 다시 응답을 기다리게 된다. 이의 경우에 직관적으로 볼 때에도 알 수 있듯이, ARQ_BLOCK_LIFETIME이 길면 ARQ 재전송이 일어나는 중간에 TCP는 재전송을 시도할 것이다. 반대로 이 시간이 너무 짧으면, ARQ를 사용함으로써 얻을 수 있는 이득이 줄어들 것이다. 이렇게 TCP를 기반으로 한 응용을 IEEE 802.16 시스템에서 사용할 때, ARQ 재전송과 TCP에서의 재전송이 서로 맞물려 동작하지 않기 때문에, 적절한 ARQ 타이머 값의 설정 없이는 TCP 성능에 심각한 저하를 초래할 수도 있다. 이 연구에서는 인터넷 상의 임의의 호스트와 TCP 통신을 하는 IEEE 802.16 단말기를 기본적인 시나리오로서 고려하기 때문에, TCP 전송 계층에 수정을 가하는 것은

부적절하다.

3.1. 송신과 관련된 ARQ 타이머

IEEE 802.16 ARQ 기능을 송신측에서 사용하는 경우에 쓰이는 타이머들은 다음과 같다.

• **ARQ_RETRY_TIMEOUT:** 송신측에서 ARQ 블록을 전송 후, 이 시간이 지날 때까지 ACK을 받지 못하면, Waiting for retransmission 상태로 들어간다. 이것은 기지국과 단말기 MAC 계층 사이에서의 round-trip 시간보다는 커야 할 것이지만, 너무 크면 재전송을 하기까지 많이 기다려야 할 수 있으므로, 성능이 나빠질 수도 있다.

• **ARQ_BLOCK_LIFETIME:** 하나의 ARQ 블록이 재전송이 가능한 상태에 머무는 최대 시간을 지정한다. ARQ 재전송 유무나 횟수에 관계없이, 이 시간이 경과하면 재전송을 포기하고, Discarded 상태로 들어간다(즉, 해당 블록을 버린다). 따라서, 최소한 ARQ_RETRY_TIMEOUT 보다는 커야만 할 것이다. 한편 이 시간을 늘리면, 이론적으로는 무한 재전송이 가능할 것이나, ARQ 자체에도 TCP처럼 전송 윈도우가 존재하기 때문에, 그 이후의 ARQ 블록들이 모두 대기 큐(queue)에 쌓인 채 진행이 되지 않을 것이다. 이는 결국 성능의 저하로 이어질 수 있으며, 동시에 TCP 송신측에서도 타임아웃을 발생시켜, TCP 프로토콜에 의한 또 다른 재전송을 초래하게 된다.

3.2. 수신과 관련된 ARQ 타이머

IEEE 802.16 ARQ 수신측인 경우에 사용하는 타이머들은 다음과 같다.

• **ARQ_RX_PURGE_TIMEOUT:** 하나의 MAC SDU는 여러 개의 ARQ 블록으로 구성되어 있음을 앞서 그림 1에서 설명하였다. 수신측에서는 MAC SDU를 재구성하기 위해 필요한 모든 ARQ 블록들을 기다리는데, 뒷부분들은 정상적으로 이미 수신되었는데도 앞 부분에 해당하는 블록들이 계속 오지 않을 수 있다. 송신측에서 ARQ_BLOCK_LIFETIME에 의해서 전송을 포기할 수 있으므로, 수신측에서도 이를 무한정 기다릴 수 없다. 이러한 경우 ARQ_RX_PURGE_TIMEOUT 만큼 기다린 뒤, 타임아웃이 발생하면 해당 SDU와 관련된 모든 블록들을 무시하고, 수신측 윈도우를 진행시킨다(즉, 해당 SDU에 대해서는 더 이상의 ARQ 동작을 포기한다). 따라서 이 값이 ARQ_BLOCK_LIFETIME 보다 작으면, 송신측에서 보낸 재전송 블록을 받지 못할 수 있고, 반대로 너무 크면, 수신측 ARQ 윈도우가 증가하지 않기 때문에 성능이 떨어진다.

• **ARQ_SYNC_LOSS_TIMEOUT:** ARQ 동작에서는 송수신 양쪽이 블록 번호에 따라 서로 적절히 동기를 유지하며 송신 윈도우를 증가시키고, 수신 윈도우가 이를

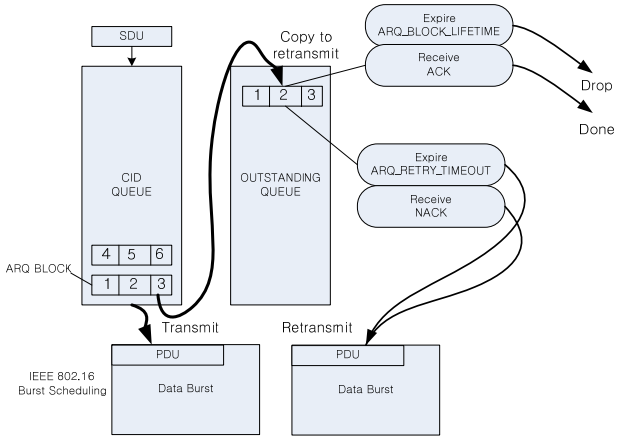

그림 3. ARQ 송신측 구현

따라간다. 하지만 신뢰성이 떨어지는 무선환경 특성에서는 다양한 이유로 이러한 동기가 깨지는 상황이 발생할 수 있다. 이럴 때에는 수신측에서 예상하는 블록 번호를 기다리기 보다는 송신측에서 새로 보내는 번호에 다시 동기를 맞추는 것이 유리하다. 따라서 ARQ_SYNC_LOSS_TIMEOUT을 기다려도 다음 블록 번호가 들어오지 않으면, 송신측이 보내는 번호로 동기를 새로 맞추게 된다. 가장 흔한 경우로서, 전송이 일시적으로 없을 때에도, ARQ_SYNC_LOSS_TIMEOUT이 발생하게 된다. 이것은 ARQ_RX_PURGE_TIMEOUT 보다 크게 정하는 것이 합리적이나, 성능 보다는 ARQ 동작의 정확성과 더 관련이 있다.

4. 시뮬레이션

4.1. 시뮬레이션 모델

본 논문에서는 NIST (National Institute of Standards and Technology)에서 배포한 IEEE 802.16 NS-2 코드 [7]를 기반으로, IEEE 802.16 물리 계층을 추상화하고, MAC 계층의 ARQ 동작을 새로 구현하였다. 1개의 MAC TDD (Time Division Duplexing) 프레임(frame)의 길이가 5ms일 때, 1대의 단말과 1대의 기지국이 서로 IEEE 802.16 네트워크 등록 절차를 완료한 상태로 가정하였으며, 상하향 링크의 connections를 성공적으로 만들었다고 가정하였다. 단말기는 이동하지 않는 대신, PDU 에러율(PDU Error Rate; PER)을 달리 하여 실험함으로써 무선 채널 상태를 변화시켰다. 단방향 하향 링크 TCP 트래픽에 대해 성능 실험을 가정하고, ARQ 역시 하향 링크에 대해서만 적용하였다. 다만, 무선 환경에서의 에러는 상하향 동시에 같은 PER을 적용하여, 하향링크 TCP 데이터뿐 만 아니라 TCP ACK이나 ARQ ACK도 무선 링크에서 오류가 날 수 있도록 하였다.

4.2. IEEE 802.16 ARQ 구현

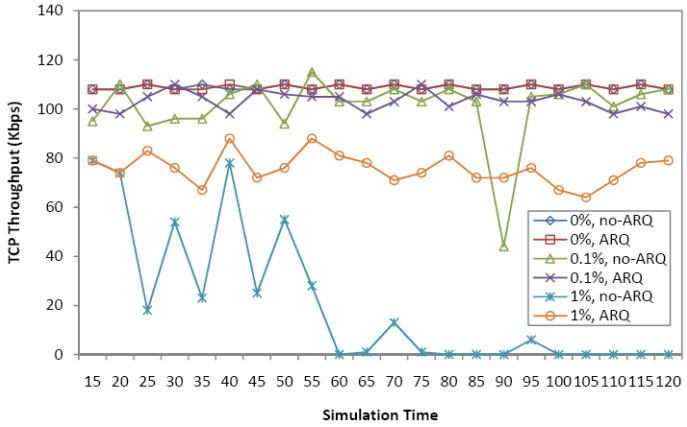

그림 4. PER에 따른 TCP 성능

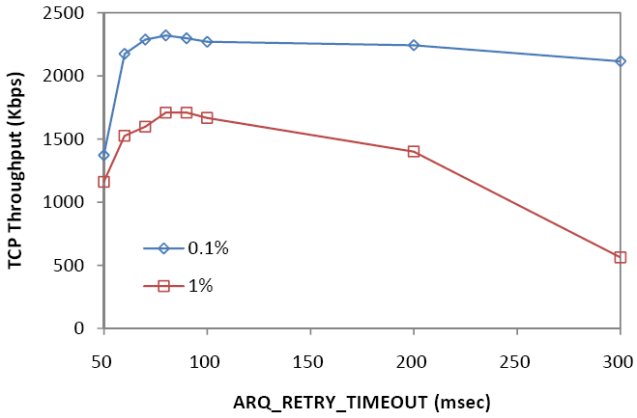

그림 5. ARQ_RETRY_TIMEOUT에 따른 TCP 성능

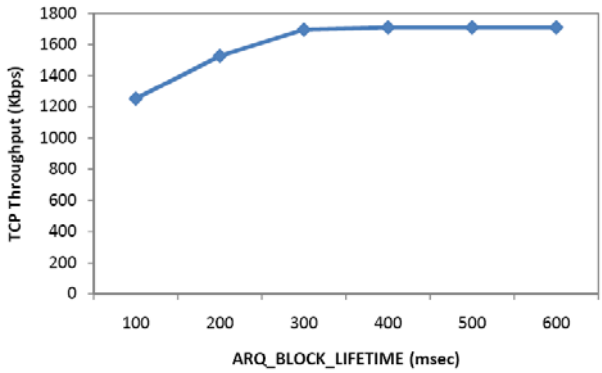

그림 6. ARQ_BLOCK_LIFETIME에 따른 TCP 성능

그림 3은 송신 측 ARQ 구현을 위해 도입한 OUTSTANDING 큐와 관련된 동작을 보여준다. 기존에 ARQ가 없는 동작에서는 CID 큐의 SDU가 바로 데이터 버스트(data burst)에 IEEE 802.16 PDU 형태로 실려 전송된다. ARQ를 도입하면, 우선 SDU들이 ARQ 블록으로 구분되고, 전송을 하는 동시에 OUTSTANDING 큐에 복사본이 저장된다. 이 순간, 각 ARQ 블록에 대하여 각종 타이머가 설정되며, ARQ_RETRY_TIMEOUT까지 ACK이 오지 않으면, 재전송을 실시한다. 또, ARQ_BLOCK_LIFETIME이 지나도록 OUTSTANDING 큐에서 제거되지 못한 블록들은 버려진다. ARQ의 ACK을 보내는 방법으로는 Cumulative ACK만 사용하였다[1]. Cumulative ACK은 그 마지막 수신이 성공한 블록 순서 번호만 전송하기 때문에, 그 오버헤드(overhead)가 매우 작고, 구현이 쉬우며 수신 측 버퍼(buffer) 요구사항이 적기 때문에 실제로 가장 널리 사용이 된다. 그 외에 ARQ 블록의 크기는 64 bytes로 하고 ARQ 전송 윈도우의 크기는 최대 128개 블록으로 정하여 실험하였다.

4.3. 시뮬레이션 결과

표 1. 시뮬레이션에 사용한 ARQ 타이머 값 (msec)

ARQ 타이머	실험1	실험2	실험3
ARQ_RETRY_TIMEOUT	100	vary	80
ARQ_BLOCK_LIFETIME	1000	1000	vary
ARQ_RX_PURGE_TIMEOUT	1000	1000	1000
ARQ_SYNC_LOSS_TIMEOUT	1500	1500	1500

그림 4은 PER이 0%, 0.1%, 1%일 때 ARQ를 사용한 경우와 그렇지 않은 경우 각각에 대해 TCP Throughput을 보여준다. ARQ가 동작하는 경우에 한해 각 타이머의 설정은 표 1에 명시한 실험 1의 경우에 준한다. 에러가 없는 경우에는, 두 경우가 별다른 차이가 없으나, 1%인 경우에서 ARQ를 사용하지 않으면 거의 정상적인 수신이 불가능함을 알 수 있다. 또 0.1%인 경우일지라도 ARQ를 사용하는 경우에는 100Kbps 근처에서 안정적인 수준의 Throughput을 보이지만, ARQ를 사용하지 않는 경우에는 TCP 성능이 순간적으로 크게 감소할 수도 있음을 보여준다.

그림 5는 ARQ_RETRY_TIMEOUT 값의 변화에 따른 TCP Throughput의 차이를 보여준다. 이때 사용한 타이머의 값들은 표 1에 명시한 실험 2에 나타내었다. 이 결과에 따르면, 너무 작은 ARQ_RETRY_TIMEOUT 값이 좋지도, 그렇다고 너무 큰 값이 좋지도 않은 것을 알 수가 있다. 기본적으로 기지국 단말간 1회 round-trip을 하는데 필요한 시간은 큐잉(queueing)과 스케줄링(scheduling) 시간까지 포함하여 대략 수 TDD 프레임 전송시간에 해당할 것이고, 앞서 가정한 바에 따라 5ms 프레임이라고 할 때, 수십 ms 정도가 걸릴 것으로 예상된다. 따라서 50ms 이하의 ARQ_RETRY_TIMEOUT은 ARQ 처리시간까지 고려했을 때 너무 짧은 것으로 보인다. 이것은 또 빈번한 ARQ 재전송을 야기하기 때문에, PER 0.1%인 경우 80ms에서, PER 1%인 경우 80ms 또는 90ms에서 최적의 성능을 보였다. 한편, 이 값이 100ms 이상으로 커지면 재전송을 하기까지 기다리는 시간이 늘어나면서 TCP 성능이 천천히 떨어지고, 특히 PER이 클수록 성능

저하가 심하게 나타남을 볼 수 있다.

그림 6에서는 PER 1%일 때, ARQ_BLOCK_LIFETIME에 따른 성능의 변화를 보였다. 이때 다른 타이머의 값들은 표 1의 실험 3과 같이 정하였다. 여기서 ARQ_BLOCK_LIFETIME이 증가함에 따라 TCP Throughput이 증가하다가 300ms 이상이 되면 더 이상 증가하지 않는 것을 볼 수 있다. 직관적으로 볼 때, ARQ_BLOCK_LIFETIME에 비례하여 재전송이 늘기 때문에 Throughput이 더 증가해야 하지만, TCP도 자신의 타임아웃이 지나면 재전송을 시작하게 되고, ARQ 전송 윈도우 안에 수용할 수 있는 패킷은 한계가 있기 때문에 (실험에서는 128개 ARQ 블록), 일정 수준이 지나면 더 이상 성능이 증가하지 않는다. 일반적으로 TCP 타임아웃은 Round-trip time (RTT)의 2배로 정해진다[8][9]. 또 앞의 실험을 통하여 80ms 정도면 충분히 Round-trip을 하고도 남는 시간이기 때문에, 대략 이것의 두 배 수준에서 TCP 타임아웃이 정해졌을 것이라 예측이 가능하다. 따라서, 성능이 증가하는 300ms 정도의 ARQ_BLOCK_LIFETIME이 적절하며, 그보다 큰 값은 ARQ 재전송을 더 하는 동시에 TCP 재전송을 야기시킬 것이므로 더 이상의 성능 증가를 기대하기 힘들다.

한편 ARQ_RX_PURGE_TIMEOUT은 1000ms로 설정하여 ARQ_BLOCK_LIFETIME과 같게 하였는데, 이것의 절대적인 값 보다는 ARQ_BLOCK_LIFETIME과 같게 설정하는 것이 나은 성능을 보였다. 또, ARQ_SYNC_LOSS_TIMEOUT은 정상적인 송수신 실험에서는 별다른 영향을 미치지 못했다(다만, ARQ_RX_PURGE_TIMEOUT보다는 커야 한다). 하지만, 여기서 실험하지 않은 기지국간의 핸드오버(handover) 상황이나 그 외의 다양한 이유로 양쪽 ARQ 상태의 동기가 서로 맞지 않는 경우에 있어, TCP 성능을 끌어올리는데 역할을 할 수 있다.

5. 결 론

본 논문에서는 IEEE 802.16 기반 시스템에서 ARQ를 사용할 때, 타이머 값들에 따라 TCP 성능 변화를 보이고, 적절한 값들을 제시하였다. 우리는 먼저 IEEE 802.16 시스템에서의 ARQ 동작을 설명하고, 어떠한 타이머들이 있고, 어떤 측면에서 각 타이머들의 값을 고려해야 하는지 살펴보았다. 또 기존의 IEEE 802.16 시뮬레이션 코드에 ARQ 기능을 추가로 설계하고, 어떻게 구현하였는지 설명하였다. 시뮬레이션 결과를 통해서는 먼저 채널 환경이 나쁠 때 ARQ를 사용함으로써 TCP 성능이 현저히 증가함을 보였다. 한편 최적의 성능을 얻기 위해서는 ARQ 타이머 값들의 적절한 범위를 알아내는 것이 중요한데, ARQ 송신 측에서 사용하는 ARQ_RETRY_TIMEOUT은 80-90ms 정도가 적절하며, ARQ_BLOCK_LIFETIME은 300ms가 이상적임을 보였다. 또, 수신 측 ARQ 타이머들은 비교적 TCP 성능에 미치는 영향은 낮으며, 대신 ARQ 동작의 무결성에 더 초점을 맞추어져 있음을 확인하였다. 그에 따라 ARQ_RX_PURGE_TIMEOUT은 ARQ_BLOCK_LIFETIME과 동일하게 설정하는 것이 좋으며, ARQ_SYNC_LOSS_TIMEOUT은 ARQ_RX_PURGE_TIMEOUT 보다 약간 크게 설정하는 것이 적절하다고 결론을 지었다.

6. 참고 문헌

[1] “IEEE Standard for Local and Metropolitan Area Networks--Part 16: Air Interface for Fixed Broadband Wireless Access Systems,” IEEE Std. 802.16-2004.

[2] “IEEE Standard for Local and Metropolitan Area Networks--Part 16: Amendment for Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1,” IEEE Std. 802.16e-2005 and IEEE Std. 802.16-2004/Cor 1-2005.

[3] W. R. Stevens, “TCP/IP Illustrated Volume 1,” Addison-Wesley, 1994.

[4] S. Kopparty, S. V. Krishnamurthy, M. Faloutsos, and S. K. Tripathi, “Split TCP for Mobile Ad Hoc Networks,” Global Telecommunications Conference, November 2002.

[5] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz, “Improving TCP/IP Performance over Wireless Networks,” in Proc. 1st ACM Conf. on Mobile Computing and Networking, November 1995.

[6] D. Barman, I. Matta, E. Altman, and R. E. Azouzi, “TCP Optimization through FEC, ARQ and Transmission Power tradeoffs,” in Proc. 2nd Conf. on Wired/Wireless Internet Communications, February 2004.

[7] NIST IEEE 802.16 ns-2 code, http://www.antd.nist.gov/seamlessandsecure/doc.html

[8] V. Jacobson, “Congestion Avoidance and Control,” in Proc. of SIGCOMM ’88, August 1988.

[9] D. Clark, “Window and Acknowledgement Strategy in TCP,” RFC 813, July 1982.