한국시뮬레이션학회 논문지
Vol. 23, No. 4, pp. 65-74 (2014. 12)
http://dx.doi.org/10.9709/JKSS.2014.23.4.065
ISSN 1225-5904

IEEE 802.11n 무선 랜에서 재전송 프레임 수를 줄이기 위한 향상된 Block ACK 방법

이현웅 · 김선명*

Efficient Block ACK Scheme for Reducing the Number of Retransmitted Frames in IEEE 802.11n Wireless LANs

Hyun-woong Lee · Sunmyeng Kim*

ABSTRACT

IEEE 802.11n standard has introduced the new schemes in MAC and PHY layers to improve network throughput. Frame aggregation and Block ACK are mainly defined to increase the efficiency of the MAC layer. There exists still problem in IEEE 802.11n. When block ACK request and/or response frames are missing or received in error, the sender does not know the status (success/failure) of each frame in the aggregated large frame and retransmits all the frames. This can cause a lower network performance. To solve this problem, we propose a new effective scheme, called reduced retransmission of MPDUs (RRM) scheme. In the proposed scheme, when a sender does not receive a block ACK response frame, it just transmits a next data frame and requests a block ACK. Therefore, it can retransmits the erroneous frames. Performance of the proposed scheme is investigated by simulation. Our results show that the proposed scheme is very effective and improves the performance under a wide range of channel error conditions.

Key words : Block Ack, IEEE 802.11n, Frame Aggregation, MAC

요 약

IEEE 802.11n 표준은 네트워크 성능을 향상시키기 위해 MAC과 물리 계층에서 새로운 방법들을 제안하였다. MAC 계층에서 성능 향상을 위해 제안된 주요 방법은 프레임 집적(Frame Aggregation)과 Block ACK이다. IEEE 802.11n 표준에도 여전히 문제점은 존재한다. Block ACK 요청 프레임이나 Block ACK 응답 프레임이 손실되거나 에러가 포함되어 수신되면, 전송 단말은 집적된 큰 프레임에 포함된 작은 프레임들의 성공적인 전송 여부를 알지 못하기 때문에 모든 작은 프레임을 재전송한다. 이는 성공적으로 전송된 프레임도 재전송될 수 있기 때문에 네트워크의 성능 저하를 초래할 수 있다. 이 문제를 해결하기 위해 본 논문에서는 RRM(Reduced Retransmissions of MPDUs) 방법을 제안한다. 제안된 방법에서 송신 단말이 Block ACK 응답을 못 받으면 모든 프레임을 재전송하는 대신에 다음 데이터 프레임 하나를 전송하고 다시 Block ACK를 요청한다. 응답을 받은 후에 에러가 발생한 프레임에 대해서만 재전송을 수행한다. 제안된 방법의 성능을 시뮬레이션을 통해 분석한다. 시뮬레이션 결과, 제안된 방법이 다양한 패킷 에러 환경에서 효과적이고 네트워크 성능을 향상 시키는 것을 보여주었다.

주요어 : Block Ack, IEEE 802.11n, MAC, 프레임 집적

*이 연구는 금오공과대학교학술연구비에 의하여 지원된 논문임.

Received: 22 November 2014, **Revised:** 10 December 2014, **Accepted:** 17 December 2014

***Corresponding Author:** Sunmyeng Kim
E-mail: sunmyeng@kumoh.ac.kr
Department of Computer Software Engineering, Kumoh National Institute of Technology

1. 서 론

IEEE 802.11n은 기존 무선 랜 표준인 IEEE 802.11의 성능을 향상시키기 위해 개정된 표준이다[1]. MAC 계층(MAC Layer)에서의 성능 향상은 프레임 집적(Frame Aggregation) 기술을 통해 이루어진다. 프레임 집적은 데이

터를 전송할 때, 두 개 이상의 데이터 프레임을 한 번에 전송함으로써 성능을 향상시킬 수 있는 IEEE 802.11e[2]와 802.11n 무선 랜의 주요 특징이다. IEEE 802.11 장치가 전송하는 모든 프레임은 라디오 레벨 헤더, MAC 프레임 필드 정보, IFS(Inter-Frame Spacing) 시간, 전송된 프레임에 대한 ACK 등과 같은 상당한 오버헤드(Overhead)를 갖는다. 최고 전송 속도로 데이터를 전송할 때, 이와 같은 오버헤드는 페이로드(Payload)보다 더 많은 채널 대역폭을 이용한다. 따라서 성능 저하를 초래하게 된다. 이 문제를 해결하기 위해, IEEE 802.11n 표준은 MSDU (MAC Service Data Unit) 집적과 MPDU (MAC Protocol Data Unit) 집적이라는 두 종류의 프레임 집적 방법을 정의하였다. 이 두 방법 모두 여러 개의 데이터 프레임을 하나의 큰 프레임으로 만든다. 일반적으로 한 프레임 당 하나의 제어 정보(오버헤드)가 필요하다. 프레임 집적은 여러 개의 데이터 프레임을 하나의 큰 프레임으로 만들어 여러 개의 프레임을 각각 전송하는 데 필요한 오버헤드를 줄임으로써 성능을 향상시킨다.

A-MSDU는 여러 개의 데이터 프레임을 하나의 큰 프레임으로 만든다. 큰 프레임에 포함된 모든 서브 프레임은 하나의 MAC 헤더를 공유하기 때문에 모두 같은 수신자 주소(Receiver Address)와 AC(Access Category)를 가져야 한다. A-MSDU 내의 각 MSDU는 서브 프레임 헤더에 의해 구분된다. A-MSDU는 공통의 MAC 헤더에 하나의 FCS(Frame Check Sequence)만 갖기 때문에 서브 프레임 중에 하나라도 에러가 발생하면 모든 서브 프레임이 재전송된다.

A-MPDU에 있는 모든 서브 프레임은 자신의 FCS를 갖는다. 서브 프레임에 에러가 발생하면 에러가 발생한 서브 프레임만 재전송된다. 따라서 A-MPDU 방법이 A-MSDU 방법보다 채널 에러가 높은 환경에서 더 좋은 성능을 갖는다. 에러 환경에서 각각의 서브 프레임에 대해 성공적인 수신 여부를 응답하기 위해 Block ACK 방법이 사용된다. Block ACK 방법의 동작에 대해서는 2.1절에서 설명한다. 에러 환경에서 A-MPDU 방법이 A-MSDU 방법보다 성능이 좋기 때문에 이 논문은 A-MPDU에 대해 초점을 맞춘다.

IEEE 802.11n과 관련된 기존 논문에서는 MAC 계층에서의 향상된 성능에 대한 분석에 초점을 맞췄다[3-14]. Saif 등은 A-MSDU의 효율 및 지연에 대한 영향을 분석하였다[3]. Ginzburg 등은 TCP와 UDP 패킷을 전송할 때의 A-MSDU와 A-MPDU의 영향을 분석하였다[4]. Daldoul 등은 멀티캐스트 트래픽에 대한 프레임 집적 방법의 모델을 제시하고 성능을 분석하였다[5]. Charfi 등은 실시간 응용 서비스에 대한 프레임 집적의 영향을 분석하였다[6]. Arif 등은 A-MPDU와 Block ACK의 효율 분석을 하였다[7].

IEEE 802.11n은 다음과 같은 두 가지 문제점을 갖고 있다. 첫 번째, Block ACK 방법은 전송 단말이 수신 단말로부터 Block ACK 프레임을 수신하고 에러로 표시된 MPDU들에 대해서만 재전송을 수행한다. 그러나 전송 단말이 전송한 Block ACK 요청 프레임 또는 수신 단말이 전송한 Block ACK 프레임에서 에러가 발생하면 전송 단말은 어떤 MPDU에서 에러가 발생하였는지 알 수가 없다. 그래서 전송 단말은 타임아웃(Timeout)된 후에 A-MPDU에 포함된 모든 MPDU를 재전송한다. 이는 전송에 성공한 MPDU에 대해서도 재전송이 이루어져 성능 저하를 초래한다. 두 번째, 채널 에러 환경에서는 하나의 A-MPDU가 많은 MPDU를 포함하는 경우 재전송해야하는 MPDU의 수가 증가하게 되어 네트워크 성능 저하를 초래할 수 있다.

IEEE 802.11n의 문제점을 해결하기 위해 본 논문에서는 RRM(Reduced Retransmissions of MPDUs) 방법을 제안한다. 제안하는 방법은 두 가지로 구성된다: Block Ack 재요청 방법, 동적 MPDU 수 조절 방법. Block Ack 재요청 방법에서는 위에서 언급한 첫 번째 문제점을 해결하기 위해 타임아웃이 발생하면 다음에 전송될 하나의 MPDU 데이터를 전송하고 다시 Block ACK 요청 프레임을 전송한다. 즉, 타임아웃이 발생할 때 A-MPDU에 집적된 모든 MPDU를 재전송하지 않고 다시 수신 단말에게 Block ACK를 요청하고 이에 대한 응답을 받음으로써 실제 에러가 발생한 MPDU에 대해서만 재전송을 시도함으로써 성능 향상을 가져올 수 있다. 동적 MPDU 수 조절 방법에서는 두 번째 문제를 해결하기 위해 채널 에러 환경에 따라 동적으로 A-MPDU에 포함되는 MPDU 수를 조절하는 방법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 Block ACK의 동작 원리를 간단하게 소개하고 3장에서 제안하는 RRM 방법에 대하여 자세히 기술한다. 4장에서 시뮬레이션을 통하여 제안된 방법의 성능을 분석하고 5장에서 결론을 맺는다.

2. 관련 연구

2.1 Block ACK

Block ACK는 원래 성능 향상을 위해 IEEE 802.11e MAC의 TXOP 메커니즘에서 사용하였다. Block ACK는

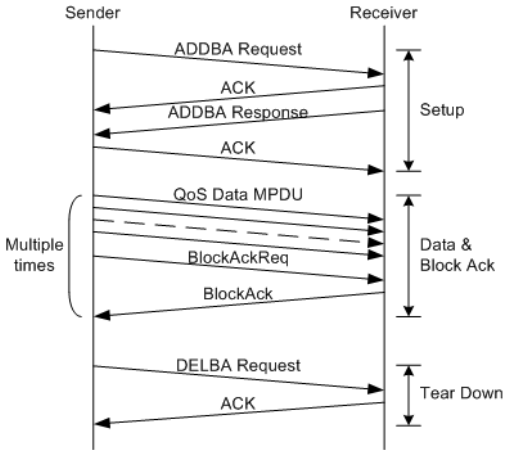

Fig. 1. Operation procedure of Block ACK

프레임 집적과 함께 IEEE 802.11n에 적용되었다. 비록 집적된 큰 프레임은 전송에서 오버헤드를 줄이지만 프레임 크기가 증가할수록 프레임 에러 확률이 커지게 되어 재전송이 더 많아진다. 따라서 에러 환경에서는 오히려 성능 저하가 초래된다. 프레임 집적의 이 문제점을 해결하기 위해 IEEE 802.11e에서 사용되던 Block ACK 방법이 수정되어 IEEE 802.11n에 적용되었다. Block ACK 방법은 A-MPDU에 있는 여러 개의 MPDU를 재전송하기 위해 사용된다. 수신 단말은 A-MPDU를 수신하고 A-MPDU에 포함된 모든 MPDU에 대해 에러 체크를 수행한다. 그리고 각 MPDU들에 대해 성공적인 수신여부를 Block ACK에 포함하여 전송 단말에게 전송한다. 전송 단말은 Block ACK 수신 후 에러가 있는 MPDU에 대해서만 재전송을 수행한다.

Block ACK 방법은 A-MSDU에는 적용되지 않고 A-MPDU에만 적용된다. A-MSDU에는 하나의 FCS만 있기 때문에 에러가 발견되면 A-MSDU 전체가 재전송된다. 하나의 A-MPDU에 포함될 수 있는 최대 MPDU의 수는 64개이다. 이는 하나의 Block ACK의 비트맵(Bitmap)이 최대 64개만 응답할 수 있기 때문이다.

Block ACK 방법에서 프레임 교환 과정은 Fig. 1과 같이 연결 설정, 데이터 및 Block ACK 전송, 연결 종료 3단계로 구성된다. 연결 설정 단계에서는 전송 단말과 수신 단말 사이에 Block ACK 방법의 연결을 설정한다. 송신 단말은 먼저 ADDBA(Add Block ACK) Request 프레임을 수신 단말에게 전송한다. 수신 단말은 ADDBA Request 프레임에 대한 ACK를 전송하고 이어서 ADDBA Request에 대한 수락 여부를 ADDBA Response 프레임

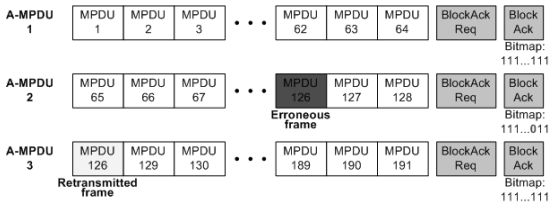

Fig. 2. Example of transmissions of A-MPDUs

으로 전송한다. 송신 단말 또한 ADDBA Response 프레임에 대해 응답 ACK를 전송한다. 이와 같은 절차를 통해 송수신 단말간 Block ACK 방법의 연결 설정이 이루어진다.

데이터 및 Block ACK 전송 단계에서는 데이터 교환이 이루어진다. A-MPDU와 Block Ack의 전송 과정은 다음과 같다. 송신 단말은 최대 64개의 MPDU들을 모아 하나의 A-MPDU를 만들고 수신 단말에게 전송한다. 수신 단말은 A-MPDU 내의 각 MPDU 프레임에 포함된 FCS를 이용하여 수신 성공/실패 여부를 체크한다. 송신 단말은 A-MPDU를 전송 완료한 후에 BlockAckReq 프레임을 수신 단말에게 전송한다. 수신 단말은 BlockAck를 송신 단말에게 전송하여 각 MPDU의 수신 성공 여부를 알린다. Block ACK 방법은 각 MPDU에 대해 성공 여부를 선택적으로 나타내기 위해 BlockAck 프레임에 Bitmap을 포함하고 있다. Bitmap 값이 1이면 해당 MPDU는 성공적인 수신을 의미하고 0이면 실패를 의미한다. 송신 단말은 Bitmap을 확인하여 전송 실패한 MPDU와 앞으로 전송 할 MPDU를 새로 집적하여 새로운 A-MPDU를 만들고 수신 단말에게 전송한다.

Fig. 2는 A-MPDU의 전송 과정 예이다. 송신 단말은 64개의 MPDU(순서번호: 1~64)를 하나의 A-MPDU로 집적하여 수신 단말에게 전송하고 이어서 BlockAckReq 프레임을 전송한다. 수신 단말은 각 MPDU에 포함된 FCS를 이용하여 성공적인 수신 여부를 파악하고 BlockAck의 Bitmap에 이 정보를 포함하여 송신 단말에게 전송한다. Fig. 2 예에서는 첫 번째 A-MPDU에 포함된 모든 MPDU를 에러없이 성공적으로 수신하였기 때문에 Bitmap은 모두 1의 값을 갖는다. 송신 단말은 다음 64개의 MPDU(순서번호: 65~128)를 A-MPDU로 만들고 전송한다. 그러나 126번 MPDU에서 에러가 발생하였다. 수신 단말은 BlockAckReq 프레임을 수신하고 126번에 해당하는 Bitmap의 값을 0으로 나타내어 BlockAck 프레임을 송신 단말에게 전송한다. 송신 단말은 BlockAck 프레임을 수신하고 Bitmap을 확인하여 126번 MPDU에서 에러가 발생한 것을 확인한다. 그리고 세 번째 전송할 A-MPDU에 재전송

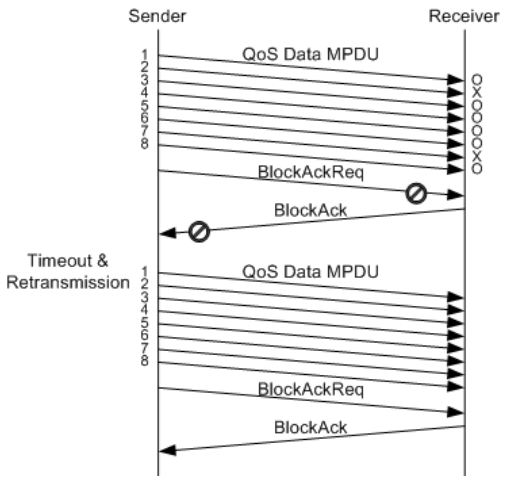

Fig. 3. Example of Block ACK problem

할 126번 MPDU와 새로운 63개의 MPDU(순서번호: 129~191)을 포함한다. 수신 단말은 모든 MPDU를 에러 없이 수신하였기에 Bitmap 값을 모두 1로 하여 BlockAck 프레임을 전송한다.

Block ACK 방법의 마지막 단계에서는 연결 종료를 수행한다. 이를 위해 송신 단말은 수신 단말에게 DELBA (Delete Block ACK) Request 프레임을 전송한다. 수신 단말은 DELBA Request 프레임에 대한 ACK를 전송하여 연결을 종료한다.

2.2 Block ACK의 문제점

Block ACK의 문제점은 BlockAckReq 프레임 또는 BlockAck 프레임의 손실에 따른 재전송 데이터의 양이 크다는 것이다. 즉, BlockAckReq 프레임 또는 BlockAck 프레임이 손실되거나 에러가 포함되어 수신하면 송신 단말에서 타임아웃이 발생하고 이전에 전송한 A-MPDU 전체를 재전송한다. 이와 같은 상황은 채널 상태가 나쁠수록 빈번하게 발생하게 되어 네트워크 성능 저하를 초래한다.

Fig. 3은 Block ACK 방법의 문제 예를 보여준다. 그림에서 A-MPDU는 8개의 MPDU로 구성된다. 송신 단말은 1~8번의 MPDU를 A-MPDU로 집적하여 수신 단말에게 전송한다. 수신 단말은 에러 있는 2번과 7번 MPDU를 수신하였고 1, 3, 4, 5, 6, 8번 MPDU는 에러 없이 수신하였다. 송신 단말은 A-MPDU 전송 후에 BlockAckReq 프레임을 송신하고 수신 단말은 해당 BlockAck 프레임을 전송한다. 이 과정에서 BlockAckReq 프레임 또는 BlockAck 프레임이 손실되거나 에러가 포함되어 있으면 송신 단말에서 타임아웃이 발생하고 응답을 못 받은 A-MPDU 전체를 재전송한다. 만약 BlockAckReq 프레임과 BlockAck 프레임에 손실되지 않거나 에러가 포함되지 않았다면 송신 단말은 에러가 포함된 MPDU 2번과 7번만 다음 A-MPDU 전송에 포함하여 재전송을 했을 것이다. 그러나 이 예에서는 1~8번의 MPDU 모두를 재전송하게 되어 네트워크 성능을 저하시킨다.

3. 제안된 RRM 방법

본 장에서는 제안하는 RRM 방법에 대해 자세히 기술한다. RRM 방법은 두 가지로 구성된다: Block Ack 재요청 방법, 동적 MPDU 수 조절 방법. 첫 번째, Block ACK 방법의 문제점을 해결하기 위해 BlockAckReq 프레임 또는 BlockAck 프레임이 손실되거나 에러가 포함되어 있어 송신 단말에 타임아웃이 발생하면 해당 MPDU를 모두 재전송하는 것이 아니라 추가적인 하나의 MPDU로 A-MPDU를 만들고 전송한다. 그리고 BlockAckReq 프레임을 전송하여 수신 단말로부터 BlockAck 프레임을 수신하여 에러가 발생한 MPDU에 대해서만 재전송을 수행하는 방법을 제안한다. 수신 단말은 이전 A-MPDU에 포함된 모든 MPDU에 대해 응답뿐만 아니라 추가적으로 전송된 하나의 MPDU에 대해서도 Block ACK에 포함하여 응답한다.

두 번째, 채널 에러 환경에서는 하나의 A-MPDU가 많은 MPDU를 포함하는 경우 재전송해야하는 MPDU의 수가 증가하게 되어 네트워크 성능 저하를 초래할 수 있다. 따라서 채널 에러 환경에 따라 동적으로 A-MPDU에 포함되는 MPDU 수를 조절하는 방법을 제안한다.

3.1 Block ACK 재요청 방법

제안된 방법은 A-MPDU, BlockAckReq 프레임, BlockAck 프레임 포맷에서 현재 사용되지 않고 예약된(Reserved) 비트를 이용하여 동작한다.

송신 단말은 BlockAckReq 프레임 또는 BlockAck 프레임의 손실 또는 에러로 인해 타임아웃이 발생하면 한 개의 MPDU만을 포함하는 A-MPDU를 전송한다. A-MPDU의 각 Sub Frame의 처음 4비트는 사용되지 않고 예약되어 있다. 이 비트들을 이용하여 송신 단말은 수신 단말에게 연속적으로 몇 개의 A-MPDU에 대해 BlockAck 프레임을 수신하지 못하였는지 알려준다. 즉, 0은 현재까지 모든 A-MPDU에 대해 에러 없는 BlockAck 프레임을 수신하였다는 것을 의미한다. 1은 현재 A-MPDU를 제외하고

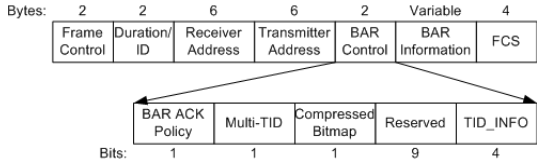

Fig. 4. Format of BlockAckReq frame

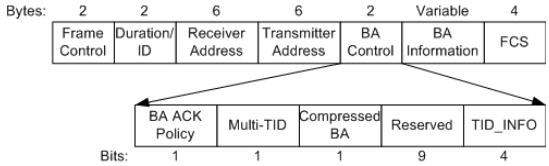

Fig. 5. Format of BlockAck frame

바로 이전 A-MPDU에 대한 BlockAck 프레임을 수신하지 못하였다는 것을 의미한다. 2는 바로 이전 두 개, 3은 세 개의 A-MPDU에 대한 BlockAck 프레임을 수신하지 못하였다는 것을 의미한다. 나머지 숫자에 대해서도 이와 같은 의미를 갖는다. 이 정보를 이용하여 수신 단말은 몇 개의 A-MPDU에 대해 수신 성공 여부 정보를 유지해야 하는지 알 수 있다. 이후부터는 이 값을 A-MPDU 인자라고 한다.

추가적인 하나의 MPDU로 구성된 A-MPDU를 전송한 후에 BlockAck 재전송 요청을 위해 송신 단말은 BlockAckReq 프레임을 전송한다. 이 프레임에는 몇 개의 A-MPDU에 대한 BlockAck를 요청하는지 나타내는 값이 포함되된다. 이 값을 나타내기 위해 BlockAckReq 프레임의 BAR Control 필드 중에서 사용되지 않는 비트들을 이용한다. Fig. 4는 IEEE 802.11n BlockAckReq 프레임의 포맷을 보여준다. Fig. 4에서 보면 BAR Control 필드 중에서 9 비트가 현재 예약된 상태이다. 이 비트 중에서 A-MPDU 인자와 같은 길이의 4비트를 이용하여 몇 개의 A-MPDU에 대한 BlockAck를 요청하는지 나타낸다. 이후부터는 이 값을 BAR 인자라고 한다. BAR 인자 값은 A-MPDU 인자 값보다 1이 크다. A-MPDU 인자 값은 현재 전송 중인 A-MPDU에 대한 정보를 포함하지 않지만 BAR 인자 값은 이를 포함하기 때문이다.

수신 단말은 BlockAckReq 프레임을 수신한 후에 BlockAck 프레임을 만들어 송신 단말에게 전송한다. BlockAck 프레임에 포함되는 Bitmap 정보는 BlockAck 프레임을 수신하지 못한 첫 번째 A-MPDU를 위해 사용된다. 두 번째 A-MPDU부터는 BlockAck 프레임의 BA Control 필드 중에서 사용되지 않는 비트들을 이용한다. 이후부터는 이 값을 BA 인자라고 한다. Fig. 5는 IEEE 802.11n BlockAck 프레임의 포맷을 보여준다. Fig. 5에서 보면 9비트가 현재 예약된 상태이다. BA 인자 값의 첫 번째 비트는 두 번째 A-MPDU의 성공적인 수신 여부를 나타내고 두 번째 비트는 세 번째 A-MPDU의 성공적인 수신 여부를 나타낸다. 세 번째 이후 비트도 이와 비슷하다. 예로 BAR 인자 값이 4이고 BA 인자 값이 11000000000이면 두 번째와 세 번째 A-MPDU에 포함된 각각의 MPDU는 성공적으로 수신한 것을 의미하고 네 번째 A-MPDU의 MPDU는 실패한 것을 의미한다. AR 인자 값이 4이므로 BA 인자 값 중에서 네 번째 비트부터는 의미 없는 0의 값을 갖는다.

Fig. 6은 BlockAckReq 프레임 또는 BlockAck 프레임의 손실 또는 에러로 인해 타임아웃이 발생한 경우의 예를 보여준다. 송신 단말은 8개의 MPDU로 구성된 A-MPDU를 수신 단말에게 전송한다. 수신 단말은 1, 3, 4, 5, 6, 8번 MPDU를 에러 없이 수신하였다. 송신 단말에서 BlockAckReq 프레임에 대한 BlockAck 프레임을 수신하지 못해 타임아웃이 발생하였다. 따라서 송신 단말은 MPDU 9번을 포함하는 새로운 A-MPDU를 수신 단말에게 전송한다. 수신 단말은 MPDU 9번을 에러 없이 수신하였다. A-MPDU에 포함된 A-MPDU 인자 값은 1이다. 전송된 A-MPDU를 수신한 수신 단말은 A-MPDU 인자 값을 확인하여 현재 A-MPDU와 바로 이전 한 개의 A-MPDU에 대한 성공적인 수신 여부 정보를 유지한다. 송신 단말은 A-MPDU 전송 후에 BlockAckReq 프레임

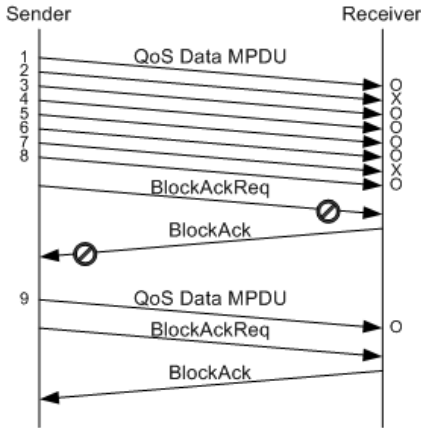

Fig. 6. Example of timeout

을 전송한다. 이 프레임에 포함된 BAR 인자 값은 2이다. 바로 이전 A-MPDU와 현재 전송 중인 A-MPDU에 대한 BlockAck 프레임을 수신하지 못했기 때문이다. BlockAckReq 프레임을 수신한 수신 단말은 BAR 인자 값을 통해 현재와 바로 이전 A-MPDU에 대한 응답이 필요한 것을 인지하고 Bitmap과 BA 인자 값을 구성하여 BlockAck 프레임을 송신 단말에 전송한다. Bitmap은 처음 전송된 A-MPDU에 포함된 각 MPDU에 대한 성공적인 수신 여부 정보를 포함하고 BA 인자 값은 현재 A-MPDU에 대한 성공적인 수신 여부 정보를 포함한다. 즉, Bitmap 값은 10111101이고 BA 인자 값은 100000000이다. BA 인자 값 중에서 첫 번째 비트만 의미 있고 나머지 비트는 의미 없는 값이다.

만약 Fig. 6에서 두 번째 BlockAckReq 프레임에 대해서도 타임아웃이 발생하면 송신 단말은 MPDU 10번을 포함하는 A-MPDU를 수신 단말에게 전송한다. A-MPDU에 포함된 A-MPDU 인자 값은 2이다. 수신 단말은 MPDU 10번을 에러 없이 수신하였다고 가정한다. 송신 단말은 BlockAckReq 프레임을 전송한다. 여기에 포함된 BAR 인자 값은 3이다. 수신 단말은 BlockAckReq 프레임을 수신한 후에 Bitmap(10111101)과 BA 인자 값(110000000)을 구성하여 BlockAck 프레임을 송신 단말에 전송한다.

3.2 동적 MPDU 수 조절 방법

채널 에러 환경에서 하나의 A-MPDU에 포함되는 MPDU의 수를 동적으로 결정하는 방법을 설명한다.

MPDU의 수는 BlockAck 프레임의 연속적인 수신 성공 또는 실패에 따라 결정된다. 하나의 A-MPDU에 포함되는 MPDU의 수($No.MPDU$)는 최대 64개에서 최소 2개로 다음과 같이 결정된다.

$$No.MPDU = \frac{m}{2^n} \tag{1}$$

여기에서 m은 하나의 A-MPDU가 포함할 수 있는 최대 MPDU 수이다. 즉, 64이다. 그리고 n은 0에서 5사이의 정수 값을 갖는 조절 변수이다. 조절 변수의 초기 값은 0이다.

일반적인 A-MPDU는 2개 이상의 MPDU로 구성된다. 한 개의 MPDU로 구성된 A-MPDU는 3.1절에서 설명한 것처럼 BlockAckReq 프레임에 대한 타임아웃이 발생한 이후 전송되는 프레임이다. 일반적인 A-MPDU를 집적할 때 조절 변수 n의 값을 결정하고 Eq. (1)을 이용하여 $No.MPDU$ 값을 결정한다. 즉, 타임아웃 이후에 전송되는 A-MPDU를 집적할 때는 조절 변수를 고려하지 않는다.

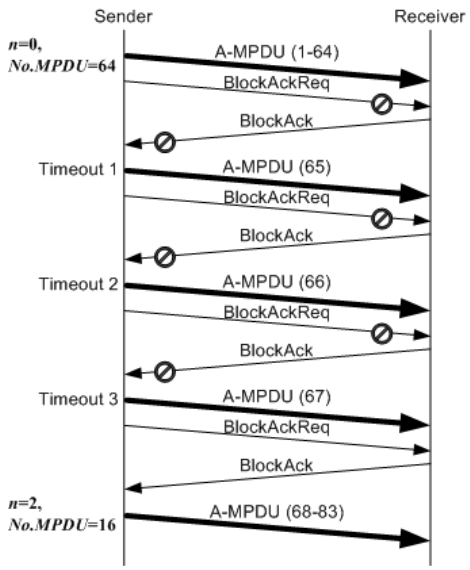

Fig. 7. Increment of adjustment parameter value

조절 변수 n은 다음 두 가지 상황에 따라 값이 증가 또는 감소한다. 일반적인 A-MPDU 프레임 전송 후에 연속적으로 발생한 타임아웃 횟수를 T라 하자. 송신 단말은 BlockAck 프레임을 에러 없이 수신하고 다음에 전송할 새로운 A-MPDU를 집적할 때, 조절 변수 값을 다음과 같이 갱신된다.

$$n = n + (T - 1) \tag{2}$$

여기에서 -1은 채널 에러 환경에 대한 마진을 부여하기 위해서이다. 송신 단말은 BlockAck 프레임을 수신하면 T 값을 0으로 설정한다. 타임아웃 이후에 전송된 A-MPDU가 아닌 일반적인 A-MPDU에 대해 연속적으로 BlockAck 프레임을 에러 없이 수신할 때마다 조절 변수 값은 다음과 같이 갱신된다.

$$n = n - 1 \tag{3}$$

타임아웃이 발생한 이후에 전송된 BlockAck 프레임의 수신에 대해서는 고려하지 않는다.

Fig. 7은 조절 변수 값의 증가 예를 보여준다. 그림에서 괄호에 있는 숫자는 해당 A-MPDU에 포함된 MPDU

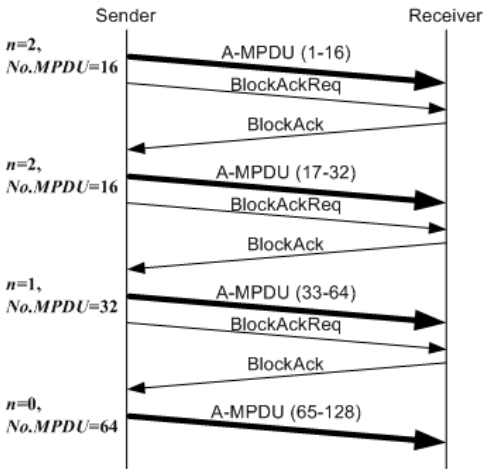

Fig. 8. Decrement of adjustment parameter value

의 순서번호이다. 그림에서 초기 n의 값은 0, 전송된 모든 MPDU는 에러 없이 수신 단말이 수신한다고 가정한다. 그림에서 송신 단말은 $No.MPDU$ 값이 64이므로 64개의 MPDU(순서번호: 1~64)를 집적하여 첫 번째 A-MPDU를 수신 단말에게 전송한다. BlockAckReq 프레임 또는 BlockAck 프레임의 손실 또는 에러로 인해 타임아웃이 발생(Timeout 1)하였다. 따라서 65번 MPDU만을 포함하는 새로운 A-MPDU를 수신 단말에게 전송하고 BlockAckReq 프레임을 전송한다. 이 또한 타임아웃이 발생(Timeout 2)하였다. 송신 단말은 66번 MPDU의 A-MPDU를 전송하였고 타임아웃이 다시 발생(Timeout 3)하였다. 그리고 67번 MPDU의 A-MPDU를 전송하였고 이번에는 수신 단말로부터 에러 없는 BlockAck 프레임을 수신하였다. BlockAck 수신 후에 송신 단말은 새로운 A-MPDU에 포함될 MPDU의 수($No.MPDU$)를 결정한다. 타임아웃이 연속적으로 3번 발생하였기 때문에 $T=3$이다. 새로운 조절 변수의 값은 $n=0+(3-1)=2$이고 MPDU 수는 $No.MPDU=64/4=16$이다. 다음에 전송할 A-MPDU는 16개 MPDU(순서번호: 68~83)로 구성된다.

Fig. 8은 조절 변수 값의 감소 예를 보여준다. 그림에서 초기 n의 값은 2, 전송된 모든 MPDU는 에러 없이 수신 단말이 수신한다고 가정한다. 송신 단말은 새로운 A-MPDU를 전송하기 전에 조절 변수 n의 값과 $No.MPDU$ 값을 결정한다. Fig. 8에서는 초기 n의 값이 2이므로 $No.MPDU$ 값은 16이다. 따라서 송신 단말은 16개의 MPDU (순서번호: 1~16)를 집적하여 첫 번째 A-MPDU를 수신 단말에게 전송한다. BlockAckReq 프레임과 BlockAck 프레임에 손실 또는 에러가 발생하지 않아 송신 단말은 BlockAck를 잘 수신하였다. 따라서 두 번째 A-MPDU를 전송한다. 두 번째 A-MPDU를 전송하는 시점에서 하나의 A-MPDU에 대해서만 에러 없이 성공하였기 때문에 연속적인 성공이 아니다. 따라서 n의 값은 감소하지 않는다. 즉, n은 2, $No.MPDU$는 16으로 변함이 없다. 따라서 두 번째 A-MPDU는 순서번호 17~32 MPDU로 구성된다. 두 번째 A-MPDU에 대한 BlockAck 프레임도 잘 수신하였다. 세 번째 A-MPDU를 구성할 때 이전에 연속적으로 전송 성공(A-MPDU(1-16)와 A-MPDU(17-32))이 이루어졌으므로 n 값과 $No.MPDU$ 값은 새롭게 결정된다. 즉, n은 1이고 $No.MPDU$는 32이다. 따라서 A-MPDU는 순서번호 33~64 MPDU로 구성된다. 이에 대한 BlockAck 프레임 수신도 잘 하였다. 네 번째 A-MPDU를 집적할 때에도 연속적인 전송 성공(A-MPDU(17-32)와 A-MPDU(33-64))이므로 n은 0, $No.MPDU$는 64가 된다. 마지막 전송의 A-MPDU는 순서번호 65~128 MPDU로 구성된다.

4. 시뮬레이션 결과

본 장에서는 제안된 방법의 성능을 IEEE 802.11n 방법과 시뮬레이션을 통해 비교 분석한다. 시뮬레이션에서 사용된 전송 속도는 150Mbps이고 MPDU 데이터 크기는 4085바이트이다. 전파 지연(Propagation Delay)은 없는 것으로 가정하였다. 시뮬레이션에서 제안된 방법의 성능에 대해서만 분석하고 다른 요소는 배제하기 위해 전송 단말 수는 한 개인 환경을 고려하였다. 즉, 데이터 전송에서 충돌은 발생하지 않는다.

IEEE 802.11n Block ACK 방법과 본 논문에서 제안된 RRM Block ACK 방법에 대해 패킷 에러 율(Packet Error Rate)에 따른 성능을 분석하였다. 100만 개의 MPDU 데이터를 수신 단말에게 성공적으로 전송할 때까지의 성공 BlockAck 수, 실패 BlockAck 수, 재전송된 MPDU 수, 전송 시간 등의 성능 요소를 고려하였다.

- 성공/실패 BlockAck 프레임 수(Number of Success/Failure BlockAck Frames): 송신 단말이 BlockAckReq 프레임을 전송하고 BlockAck 프레임을 에러 없이 수신하면 성공이고 BlockAck 프레임을 수신하지 못하거나 에러 있는 프레임을 수신하면 실패이다.
- 재전송된 MPDU 수(Number of Retransmitted MPDUs): 송신 단말이 전송한 MPDU 중에서 에러

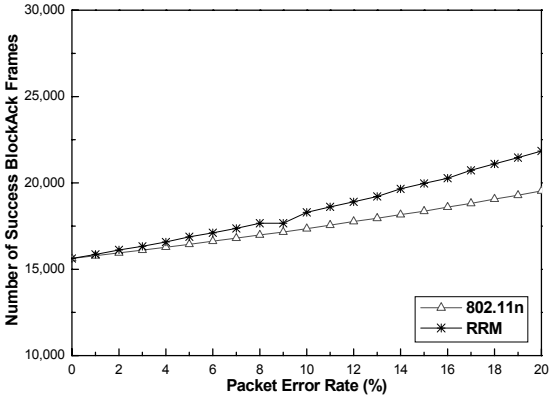

Fig. 9. Number of success BlockAck frames according to the packet error rate

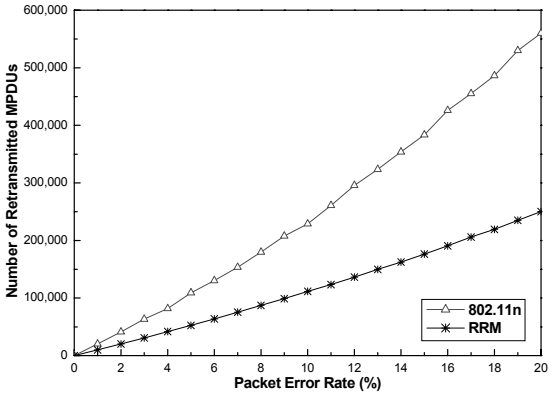

Fig. 11. Number of retransmitted MPDUs according to the packet error rate

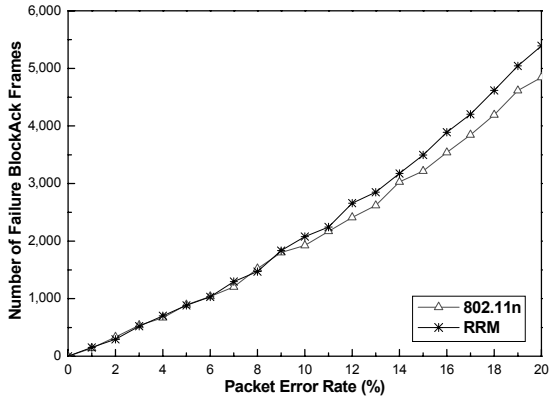

Fig. 10. Number of failure BlockAck frames according to the packet error rate

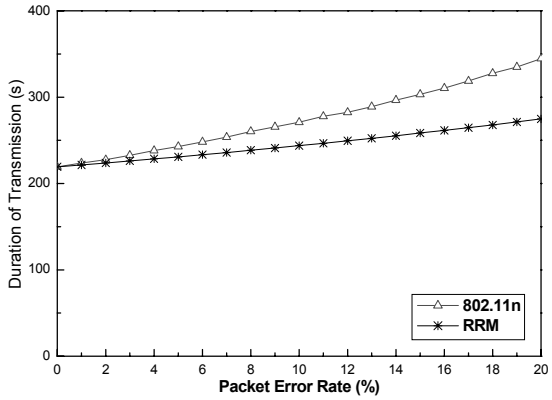

Fig. 12. Duration of transmission according to the packet error rate

가 발생하거나 BlockAck 프레임 수신 실패로 인해 재전송된 총 MPDU 수를 의미한다.

- 전송 시간(Duration of Transmission): 100만 개의 MPDU를 전송하는데 소요된 총 시간을 나타낸다.

Fig. 9는 패킷 에러 율에 따른 100만 개의 MPDU를 전송 완료할 때까지 성공적으로 수신된 BlockAck 프레임 수를 나타낸다. 패킷 에러 율이 0%일 때, 각 A-MPDU는 64개의 MPDU로 구성된다. 따라서 전송된 총 A-MPDU 수는 15625개(=1,000,000 / 64)이고 BlockAck 프레임 수도 15,625개이다. 패킷 에러 율이 증가함에 따라 MPDU의 에러 또는 BlockAck 프레임의 수신 실패로 인한 MPDU들의 재전송도 증가한다. 따라서 전송된 총 BlockAck 프레임 수도 증가하게 되어 성공 BlockAck 수도 점점 커진다. 그림에서 제안된 RRM 방법이 IEEE 802.11n 방법보다 성공 BlockAck 수가 좀 더 빠르게 증가함을 볼 수 있다. 이는 제안된 방법에서 일반적인 A-MPDU에 대한 BlockAck 프레임의 수신 실패로 인해 추가적인 하나의 MPDU로 구성된 A-MPDU를 전송하고 BlockAckReq 프레임을 전송하기 때문이다.

Fig. 10은 패킷 에러 율에 따른 실패 BlockAck 프레임 수를 나타낸다. 패킷 에러 율이 0%일 때는 실패한 BlockAck 프레임은 존재하지 않는다. 그러나 패킷 에러 율이 증가함에 따라 점점 실패 BlockAck 프레임 수는 증가한다. 그림에서 알 수 있듯이, 제안하는 RRM 방법이 IEEE 802.11n 방법보다 실패 BlockAck 프레임 수가 좀 더 빠르게 증가한다. Fig. 9의 결과와 마찬가지로 전송된 총 BlockAck 프레임 수가 많기 때문이다.

Fig. 11은 패킷 에러 율에 따른 재전송된 MPDU 수를 나타낸다. 패킷 에러 율이 증가함에 따라 제안된 RRM 방

법은 선형적으로 증가하지만 IEEE 802.11n 방법은 급격하게 증가하는 것을 볼 수 있다. 이는 IEEE 802.11n 방법에서는 에러 있는 MPDU뿐만 아니라 BlockAck 프레임 수신 실패로 인해 에러 없이 수신 단말에게 전송된 MPDU에 대해서도 재전송이 이루어지는 반면 제안된 방법은 에러 있는 MPDU에 대해서만 재전송을 하기 때문이다. 패킷 에러 율이 20%일 때, 단순 비율로는 100만 개의 MPDU 중에서 에러가 발생할 수 있는 개수는 20만개(100만 * 0.2)이다. 그러나 제안된 방법에서는 25만개의 MPDU에 대해 재전송이 이루어졌다. 이는 에러로 인해 재전송 MPDU 프레임이 다시 에러가 발생하여 또 재전송되었기 때문이다. 패킷 에러 율이 20%이면 성공 율은 80%이다. 그러므로 이 환경에서 100만 개의 MPDU를 에러 없이 수신 단말에게 전송하기 위해서는 총 125만 개(=1,000,000 / 0.8)의 전송이 필요하다. 즉, 25만 개 MPDU에 대해 재전송이 필요하다는 것을 의미한다. 그러므로 제안된 방법은 수학적으로 계산된 개수만큼의 MPDU 재전송이 이루어졌다. 따라서 오버헤드는 25%(=250,000 / 1,000,000)이다. 그러나 IEEE 802.11n 방법의 오버헤드는 56%(=560,000 / 1,000,000)이다. 이 결과는 제안된 방법이 패킷 에러 율이 높을수록 훨씬 더 좋은 성능을 가짐을 의미한다.

Fig. 12는 패킷 에러 율에 따른 전송 시간을 나타낸다. 에러가 없는 상황에서는 두 방법 모두 약 220초의 시간이 걸렸다. 패킷 에러 율이 증가함에 따라 전송 시간도 증가한다. 제안된 방법은 선형적으로 증가하지만 IEEE 802.11n 방법은 급격하게 증가함을 볼 수 있다. Fig. 11 결과에서 보인 것처럼 제안된 방법은 재전송된 MPDU 수가 적은 반면 IEEE 802.11n 방법은 재전송된 MPDU 수가 크기 때문에 이에 따른 전송 시간 차이를 유발한다.

5. 결 론

IEEE 802.11n 표준은 네트워크 성능을 향상시키기 위해 MAC 계층에서 프레임 집적과 Block ACK 방법을 제안하였다. IEEE 802.11n 표준에도 여전히 문제점은 존재한다. Block ACK 요청 프레임이나 Block ACK 응답 프레임이 손실되거나 에러가 포함되어 수신되면, 전송 단말은 집적된 큰 프레임에 포함된 작은 프레임들의 성공적인 전송 여부를 알지 못하기 때문에 모든 프레임을 재전송한다. 이는 성공적으로 전송된 프레임도 재전송될 수 있기 때문에 네트워크의 성능 저하를 초래할 수 있다. 이 문제를 해결하기 위해 본 논문에서는 새로운 방법을 제안하였다. 제안된 방법에서 Block ACK 응답을 못 받으면 다음 데이터 프레임 하나를 전송하여 Block ACK를 재요청한다. 응답을 받은 후에 에러가 발생한 프레임에 대해서만 재전송을 수행한다. 제안된 방법의 성능을 시뮬레이션을 통해 분석하였다. 시뮬레이션 결과, 제안된 방법이 다양한 패킷 에러 환경에서 효과적이고 네트워크 성능을 향상 시키는 것을 보여주었다.

References

1. IEEE Std 802.11n, “IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Enhancements for Higher Throughput,” Oct. 2009.
2. IEEE Std 802.11e, “IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC) Quality of Service Enhancements,” 2005.
3. A. Saif, M. Othman, S. Subramaniam, and N. AbdulHamid, “Impact of Aggregation Headers on Aggregating Small MSDUs in 802.11n WLANs,” 2010 International Conference on Computer Applications and Industrial Electronics (ICCAIE), pp. 630-635, Dec. 2010.
4. B. Ginzburg and A. Kesselman, “Performance Analysis of A-MPDU and A-MSDU Aggregation in IEEE 802.11n,” IEEE Sarnoff Symposium, pp.1-5, 2007.
5. Y. Daldoul1, T. Ahmed, and D. Meddour, “IEEE 802.11n Aggregation Performance Study for the Multicast,” IFIP Wireless Days(WD), pp. 1-6, Oct. 2011.
6. E. Charfi, L. Chaari, and L. Kamoun, “Fairness of the IEEE 802.11n Aggregation Scheme for Real Time Application in Unsaturated Condition,”IFIP Wireless and Mobile Networking Conference (WMNC), pp.1-8, Oct. 2011.
7. T.Y. Arif and R.F. Sari, “Throughput Estimates for A-MPDU and Block ACK Schemes Using HT-PHY Layer,” Journal of Computing, Vol. 9, No. 3, pp. 678-687, March 2014.
8. Y. Lin and V.W.S. Wong, “Frame Aggregation and Optimal Frame Size Adaptation for IEEE 802.11n WLANs,” IEEE GLOBECOM'06, pp. 1-6, Dec. 2006.
9. D. Skordoulis, Q. Ni, H. Chen, A.P. Stephens, C. Liu, and A. Jamalipour, “IEEE 802.11n MAC Frame Aggregation Mechanisms for Next-Generation High-Throughput WLANs,” IEEE Wireless Communications, Vol. 15, No. 1, pp. 40-47, Feb. 2008.
10. Y. Xiao, “IEEE 802.11n: Enhancements for Higher Throughput in Wireless LANs,” IEEE Wireless Com-

munications, Vol. 12, No. 6, pp. 82-91, Dec. 2005.
11. C.-Y.Wang and H.-Y. Wei, “IEEE 802.11n MAC Enhancement and Performance Evaluation,” Mobile Networks and Applications, Vol. 14, No. 6, pp. 760-771, Dec. 2009.
12. T.Y. Arif and R.F. Sari, “An Analytical Model of A-MSDU Scheme with Enhanced Block ACK for IEEE 802.11n Networks,” IEEE International Conference on Networks (ICON), pp. 291-298, Dec. 2012.
13. C. Shin, H. Park, and H.M. Kwon, “PHY-Supported Frame Aggregation for Wireless Local Area Networks,” IEEE Transactions on Mobile Computing, Vol. 13, No. 10, pp. 2369-2381, October 2014.
14. W.-J. Liu, C.-H. Huang, and K.-T. Feng, “Performance Analysis of Block Acknowledgement Mechanisms for Next Generation Wireless Networks,” IEEE WCNC’2010 pp. 1-6, April 2010.

이 현 웅 (daverboy89@gmail.com)

2008~현재 금오공과대학교 컴퓨터소프트웨어공학과 학사과정

관심분야 : 무선 네트워크

김 선 명 (sunmyeng@kumoh.ac.kr)

2002 아주대학교 정보통신공학과 석사
2006 아주대학교 정보통신공학과 박사
2008~현재 금오공과대학교 컴퓨터소프트웨어공학과 교수

관심분야 : 무선 네트워크