논문 2006-43TC-10-10

IEEE 802.11 기반 이동 ad-hoc 망에서 TCP 성능 향상을 위한 적응적 DCF 알고리즘 설계

(Design of Adaptive DCF algorithm for TCP Performance Enhancement in IEEE 802.11 based Mobile Ad-hoc Networks)

김 한 집*, 이 기 라*, 이 재 용**, 김 병 철**

(Han Jib Kim, Gi Ra Lee, Jae Yong Lee, and Byung Chul Kim)

요 약

TCP는 신뢰성을 보장하는 전송 프로토콜로서 인터넷 등에서 가장 널리 사용되고 있는 전송 방식이다. 하지만 TCP는 유선망에 적합하도록 설계되었기 때문에 무선망에서 TCP를 사용할 경우 성능 저하가 발생된다. TCP의 성능 저하 원인으로는 MAC 계층에서의 무선 매체 경쟁, hidden-terminal 문제와 exposed terminal 문제, 링크 계층에서의 패킷 손실, 불공정성의 문제들과 노드의 이동에 의한 경로 단절시 발생되는 패킷 순서 바뀜 문제와 경로의 단절로 인한 재전송 타이머의 exponential backoff에 의한 대역폭의 낭비 등이 있다. 특히 이동 ad-hoc 망에서는 전송 범위(transmission range)와 간섭범위(interference range)의 불일치로 인해 발생되는 hidden terminal 문제로 인해 동시에 전송할 수 있는 노드의 수가 제한되며 이로 인해 성능 저하가 크게 발생된다. 본 논문에서는 IEEE 802.11 기반 이동 ad-hoc 망에서 발생되는 hidden terminal 문제로 인해 노드가 전송을 하지 못하고 *CW*(contention window)만 크게 증가되는 문제를 해결하기 위한 MAC 알고리즘을 제안한다. 기존의 802.11 MAC의 DCF(distributed coordination function)에서는 전송에 실패할 경우 *CW* 를 지수적으로 증가 시키지만 본 논문에서 제안하는 기법은 노드가 전송 실패를 하였을 경우 그 원인에 따라 *CW* 를 적절하게 변화시킴으로 성능 향상을 얻을 수 있다. 이 기법을 사용하면 hidden terminal에 의해 전송을 실패하는 노드에게 공정한 전송 기회를 부여함으로써 TCP 성능 향상을 얻을 수 있음을 시뮬레이션을 통해 보였다.

Abstract

TCP is the most widely used transport protocol in Internet applications that guarantees a reliable data transfer. But, in the wireless multi-hop networks, TCP performance is degraded because it is designed for wired networks. The main reasons of TCP performance degradation are contention for wireless medium at the MAC layer, hidden terminal problem, exposed terminal problem, packet losses in the link layer, unfairness problem, reordering problem caused by path disconnection, bandwidth waste caused by exponential backoff of retransmission timer due to node's mobility and so on. Specially, in the mobile ad-hoc networks, discrepancy between a station's transmission range and interference range produces hidden terminal problem that decreases TCP performance greatly by limiting simultaneous transmission at a time. In this paper, we propose a new MAC algorithm for mobile ad-hoc networks to solve the problem that a node can not transmit and just increase *CW* by hidden terminal. In the IEEE 802.11 MAC DCF, a node increases *CW* exponentially when it fails to transmit, but the proposed algorithm changes *CW* adaptively according to the reason of failure so we get a TCP performance enhancement. We show by ns-2 simulation that the proposed algorithm enhances the TCP performance by fairly distributing the transmission opportunity to the failed nodes by hidden terminal problems.

Keywords: 이동 ad-hoc 망, MAC, TCP, 적응적 DCF

* 학생회원, 충남대학교 정보통신공학과
(Division of Electrical and Computer Engineering, Chungnam National University)
** 종신회원, 충남대학교 전기정보통신공학부
(Division of Electrical and Computer Engineering, Chungnam National University)
※ 본 논문은 한국과학재단의 특정기초연구 (R01-2006-000-10154-0(2006)) 지원으로 수행되었음
접수일자: 2006년4월21일, 수정완료일: 2006년10월11일

Ⅰ. 서 론

TCP는 상위계층의 서비스에 대한 신뢰성을 보장하는 전송 프로토콜로서 현재 많은 인터넷 통신 어플리케이션들이 TCP를 사용하고 있다. 하지만 TCP는 원래 유선망에 적합하도록 설계되었기 때문에 이를 무선망에 적용할 경우 성능 저하의 문제가 발생된다. 이동 ad-hoc 망에서는 무선망이 갖는 고유한 특성들 때문에 최적의 성능을 내지 못하게 되는데 이동 ad-hoc 망에서의 TCP 성능의 저하 요인으로는 MAC 계층에서의 무선 매체 경쟁, hidden-terminal 문제와 exposed terminal 문제, 링크 계층에서의 패킷 손실, 불공정성의 문제들과 노드의 이동에 의한 경로 단절시 발생되는 패킷 순서 바뀜 문제와 경로의 단절로 인한 재전송 타이머의 exponential backoff에 의한 대역폭의 낭비 등을 들 수 있다.

무선 ad-hoc 망에서 TCP를 사용할 경우 hidden terminal 문제로 인해 동시에 전송할 수 있는 노드의 수가 제한되게 된다. 이는 전송 범위와 간섭범위의 불일치로 인해 발생되는 것으로 패킷의 충돌을 일으켜 손실을 유발한다. 따라서 노드들이 동시에 전송할 수 있는 범위가 분할되는데 이를 공간재사용(spatial reuse) 성질이라 부른다. 그림 1에서는 패킷 전송 범위(250m)와 간섭 범위(550m)의 불일치로 인해 발생되는 hidden terminal 문제를 나타내었는데 이동 ad-hoc 망에서 TCP 성능 저하 원인 중에 가장 큰 문제점이다. 예를 들어 4번 노드가 5번 노드에게 패킷을 전송하는 동안 1번 노드가 2번 노드에게 패킷을 전달하면 전송에 실패하게 되는 이는 4번 노드가 2번 노드에 간섭을 일으키는 hidden terminal이기 때문이다.

IEEE 802.11 기반 이동 ad-hoc 망에서는 DCF (distributed coordination function)[4]를 사용하여 모든 노드들이 동등한 관계에서 경쟁을 통해 채널을 사용하게 된다. 이동 ad-hoc 망에서 노드는 DCF 방식을 사용할 때 RTS/CTS 기법을 사용하지만 이는 hidden terminal 문제를 해결하지 못한다. 예를 들어 그림 1에서 4번 노드가 5번 노드에게 RTS를 보내고 5번 노드가 CTS를 보내 채널 사용할 기회를 얻었어도 간섭 범위 밖에 있는 1번 노드는 이를 인지하지 못하기 때문에 2번 노드에게 전송을 시도하기 위해 RTS 패킷을 보낸다. 2번 노드는 4번 노드의 간섭 범위에 있기 때문에 1번 노드가 보낸 패킷이 충돌이 발생하여 1번 노드에게 CTS를 보낼 수 없다. RTS를 보낸 1번 노드는 2번 노드로부터 CTS를 받지 못하게 되고 timeout이 발생한다. 1번 노드는 exponential backoff를 통해 CW를 증가시키며 최대 전송횟수만큼 RTS를 보내고 CTS를 계속해서 수신하지 못하게 되면 경로 단절로 판단하고 경로 재설정을 수행하게 된다. 원래 802.11 MAC 규약에서 CW를 증가시켜 가는 이유는 RTS를 보낸 노드가 다른 노드와 충돌이 발생했다고 판단하여 충돌을 회피하기 위한 노력이다. 그러나 hidden terminal에 의한 전송실패를 충돌로 판단하여 exponential backoff를 수행하여 CW를 증가시키는 것은 트래픽 부하가 낮은 망에서는 전송시간 낭비가 되고, 또한 backoff time 후에 자신이 전송을 시도할 때 hidden terminal 문제가 또 발생하여 전송실패 가능성으로 성능 저하가 크게 나타난다. 이동 ad-hoc 망과 같이 노드의 이동이 이루어지는 경우 hidden terminal로 인한 문제가 더욱 심각하다. 이동 ad-hoc 망에서 hidden terminal 문제는 가용 대역폭 제한과 노드간의 전송 기회의 불공평성, 노드의 전송대기시간 증가, burst한 트래픽 생성, 불필요한 경로 재설정을 발생시키며 이로 인해서 TCP의 성능 저하가 발생한다[1][2][3]. 따라서 본 논문에서는 기존의 DCF에서 노드가 전송을 실패하였을 경우 CW를 지수적으로 증가시키는 방법을 개선하여 hidden terminal 문제가 존재하는 이동 ad-hoc 망환경에 적합하도록 CW를 전송실패 원인에 따라 적절하게 변화시키는 Adaptive DCF알고리즘을 제안한다.

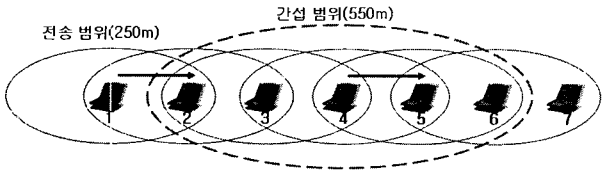


그림 1. 이동 ad-hoc 망에서 hidden terminal 문제
Fig. 1. Hidden terminal problem in mobile ad-hoc networks.

본 논문의 구성은 다음과 같다. 제 Ⅱ장에서는 기존의 IEEE 802.11 MAC 프로토콜인 DCF의 문제점에 대해서 설명하며, 제 Ⅲ장에서는 DCF를 이동 ad-hoc 망 환경에 맞도록 개선한 Adaptive DCF를 제안하고, 제 Ⅳ장에서는 이를 시뮬레이션을 통해 성능분석을 보였고, 마지막으로 제 Ⅴ장에서 결론 및 향후 연구 계획을 제시한다.

II. 기존의 IEEE 802.11 DCF 알고리즘 문제점

DCF는 IEEE 802.11 MAC[4]의 기본적인 매체 접근 방식으로서, CSMA/CA방식을 사용한다. CSMA/CA는 각 노드간의 충돌을 줄이기 위해서 random backoff time을 사용하고 있다[5][6]. 그림 2는 DCF 환경에서 노드의 동작을 보여준다. Busy medium 상태가 끝나고 DIFS 동안 매체가 유휴 상태이면, random backoff time을 생성하여 매체에 대한 접근을 연기한다. 매체에 대한 접근을 연기한 노드들은 매체의 상태를 확인하면서 동시에 자신의 random backoff time을 감소시켜 나간다. 만약 어떤 노드의 backoff time이 0이 될 때까지 매체가 유휴 상태이면 그 노드는 매체에 접근하게 되고, 0이 되기 전에 매체를 다른 노드가 사용하게 되면 backoff time을 줄이는 것을 멈추고 다음 DIFS 후에, 남아있는 backoff time을 감소시킨다. 따라서 이 노드는 처음 random backoff time 을 생성한 노드보다 더 작은 backoff time을 가지게 될 확률이 높으므로 매체에 접근할 가능성 또한 높다. 결국 backoff 시간이 0이 되면 프레임을 전송하게 되며, ACK을 통해 프레임 전송에 대한 성공여부를 결정한다. 프레임이 성공적으로 전송되었을 경우 CW 값을 $CW_{\min}$ 값으로 감소시키며, 충돌로 감지했을 경우 CW 값을 2배로 증가시킨다. 가상 캐리어 감지(virtual carrier sense)기법은 채널에 대한 예약 정보를 알리는 방식으로, 실제 데이터를 전송하기 전에 미리 짧은 길이의 RTS(Ready To Send)와 CTS (Clear To Send)를 교환하여 채널의 예약을 알리는 방식이다.

그림 3은 DCF에서 RTS/CTS 방식에 대한 그림을 보여준다. 가장 짧은 random backoff time을 생성한 노드는 매체접근에 성공하게 되고, 먼저 RTS 프레임을 전송한다. RTS 프레임에는 데이터를 전송하고자 하는 source 노드의 주소와 NAV(network allocation vector) 설정에 사용되는 duration 필드가 포함되어 있다. RTS 프레임을 수신한 노드들 중에 목적지 노드는 RTS프레임에 대한 응답으로서 CTS 프레임을 전송하고, 나머지 노드들은 자신의 NAV를 RTS 프레임에 포함된 duration 필드의 값으로 설정한 뒤, NAV를 줄여나가면서 매체접근을 연기한다. RTS/CTS 프레임의 전송이 끝나면, 송신 노드는 데이터 전송을 시작하게 되고 수신 노드는 ACK 프레임을 전송한다. 모든 프레임에 duration 필드가 포함되어 있고, 노드들은 현재 NAV보다 더 큰 duration 필드를 수신할 경우에만 NAV를 갱신한다. NAV가 0이 되면, 노드들은 매체가 유휴 상태라고 판단하고 DIFS 동안 기다린 뒤 자신의 backoff time을 줄여나가면서 매체접근을 시도한다. DCF 방식은 RTS와 CTS 교환 충돌로 인한 전송 경로의 손실을 즉각적으로 확인할 수 있음으로 RTS/CTS 기법은 평균 패킷의 크기가 큰 환경이나 노드의 수가 많아 충돌확률이 높은 환경에서 효율적이다. 하지만 RTS/CTS 방식을 사용하여도 간섭범위가 전송범위보다 크기 때문에 발생하는 hidden terminal 문제를 해결하지 못한다. 앞에서 설명한 hidden terminal로 인해 RTS/CTS를 교환이 이루어지지 못하게 되고 노드는 일정시간동안 RTS에 대한 응답이 없는 경우 재전송을 시도하고 자신의 CW의 값을 지수적으로 증가시키고 이 값과 0 사이의 임의의 값으로 새로운 backoff timer를 설정하게 된다. 연속된 전송 실패를 할 경우 노드의 CW값은 $CW_{\max}$까지 증가된다. 반면 전송 성공한 노드는 다음 패킷을 전송하기 위해서 CW값을 $CW_{\min}$값으로 설정하게 된다. 따라서 이동 ad-hoc 망에서 전송 성공 노드와 hidden terminal 문제로 전송을 하지 못한 노드가 패킷을 보내기 위한 경쟁을 할 경우 계속해서 전송에 성공한 노드가 우선권을 갖게 될 확률이 높게 되어 한 노드에 의해서 burst한 패킷 전송이 발생하고 전송 실패 노드는 계속적으로 지연될 가능성이 높아진다. Ad-hoc 망에서 노드는 최대 재전송 횟수만큼 전송 실패를 할 경우 경로가 단절되었다고 판단하고 경로 재설정을 위

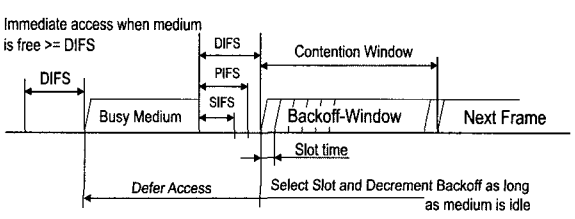


그림 2. DCF의 기본 접근 방법
Fig. 2. Basic access method in DCF.

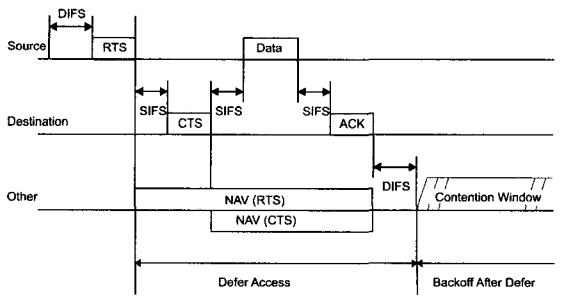


그림 3. DCF에서 RTS/CTS 접근 방식
Fig. 3. RTS/CTS access method in DCF.

한 라우팅 프로토콜이 동작하게 된다. 만약 최대 재전송 횟수만큼 전송 실패한 원인이 경로 단절이 아닌 hidden terminal이라면 경로 재설정은 불필요하다. 경로 재설정이 hidden terminal로 인해 수행된다면 경로를 찾기 위해 불필요하게 생성된 라우팅 패킷으로 인해 망이 혼잡하여지고 대역폭을 낭비함으로 인해 성능 저하가 발생된다.

따라서 본 장에서는 hidden terminal로 인해 전송에 실패한 노드의 CW가 증가되고 이로 인해 발생되는 TCP 성능 저하를 줄이기 위해 802.11 MAC DCF에서 전송에 실패하였을 경우 CW을 지수적으로 증가시키는 기법을 개선한 Adaptive DCF를 제안한다.

Ⅲ. Adaptive DCF 알고리즘

DCF 방식에서 노드가 패킷 전송을 위해 backoff timer를 감소하는 동안 다른 노드의 패킷 전송으로 인해 NAV가 설정된다. Adaptive DCF 방식은 backoff timer를 감소하는 동안 NAV 설정 횟수(NAV_{count})를 카운트하여 CW의 값을 변화시키는 방법이다. 만일 패킷간 충돌이 아니라 hidden terminal로 인해 전송 실패가 발생하였다면 노드의 CW값을 지수적으로 증가 시키는 것이 아니라 오히려 감소시킴으로 전송의 기회를 빨리 얻어 전송 기회의 공평성을 유지하고 한 노드가 계속 프레임을 전송하는 burst한 특성을 제한하려고 하는 것이다. 왜냐하면 CW를 감소시키는 도중에 한번도 프레임 전송이 감지되지 않았는데도 전송 실패가 되었다면, 프레임간의 충돌 가능성보다는 hidden terminal에 의한 원인일 가능성이 높으므로, 단순히 CW를 증가시키는 것은 합당하지 않기 때문이다.

본 기법은 패킷 전송 실패의 원인을 CW를 카운트 다운 할 때 경험한 NAV_{count}값에 따라 다르게 분류하여 그에 적합하게 CW를 증가시키거나 감소시키는 방법을 통해 hidden terminal로 인해 부적합한 CW의 증가를 막아 이동 ad-hoc 망에서 TCP 성능 감소 문제를 해결하는 것이다. 본 기법을 사용할 경우 hidden terminal을 고려하기 때문에 전송 실패한 노드의 불필요한 CW 증가를 막음으로 전송 대기 시간을 줄여 망 효율성을 증가시키고 노드간 전송 기회를 공평하게 제공함으로써 burst한 트래픽 생성을 막을 수 있다. 또한 CW 파라미터 값을 기존의 DCF에서 사용하는 값보다 작게 설정함으로 노드의 이동이 많아 망 상태 변화가 심한 이동 ad-hoc 망에 적합하도록 설정하였다. 노드는 처음 전송을 시도하거나 전송에 성공했을 때 CW_{normal}을 갖는다. 노드는 자신이 전송할 기회를 얻기 위해 대기하는 시간 동안 다른 노드가 전송하는 것을 감지하여 주변에서 전송하는 노드의 유무를 알 수 있고 이를 NAV_{count}에 누적시킨다. 따라서 자신이 패킷을 전송했을 때 실패하였다면 그 원인을 NAV를 카운트한 값 NAV_{count}를 바탕으로 판단하여 적합한 CW값으로 재설정한다. Adaptive DCF에서 패킷 전송 실패 시 그 원인을 판단하는 기준을 다음과 같이 설정한다.

◆ $NAV_{count} = 0$일 경우

다른 노드가 전송하는 것을 감지하지 못했기 때문에 이는 hidden terminal로 인해 패킷 전송이 실패한 것으로 간주하여 CW를 현재 CW에서 반으로 줄여 나가며 계속해서 hidden terminal 문제가 발생될 경우 $CW_{\min}$까지 줄인다.

◆ $1 \leq NAV_{count} \leq 2$ 일 경우

일반적인 전송 트래픽으로 인한 충돌이라 판단하고 현재 CW를 유지한다.

◆ $2 < NAV_{count}$ 일 경우

주변에 전송하는 노드가 많이 있다고 판단하여 CW를 현재 CW에서 두 배 증가시켜 충돌 확률을 줄인다. 충돌이 연속적으로 발생될 경우 $CW_{\max}$까지 증가된다.

Adaptive DCF를 사용할 경우 세 가지 CW 파라미터인 $CW_{\max}$, $CW_{\min}$, CW_{normal}을 어떻게 설정하느냐에 따라 성능의 차이가 존재한다. 기존의 DCF에서는 $CW_{\min}$을 31로, $CW_{\max}$를 1023으로 설정하고 있는데 이것은 이동성이 존재하는 이동 ad-hoc 망에서는 큰 값으로 부적합하다. 본 논문에서는 이동 ad-hoc 망에 적합한 세 가지 CW 파라미터를 다음 장에서 시뮬레이션을 통해 제안한다. 이 때 고려해야 할 사항은 노드가 전송 실패로 인해 최대 재전송 횟수만큼 시도하는 시간이 hidden terminal이 한 프레임을 전송하는데 걸리는 시간보다 작으면 안 된다는 것이다. 만약 hidden terminal이 전송하고 있는 시간 내에 노드가 최대 재전송 시도를 끝낸다면 성공적으로 전송할 수 있는 기회를 한번도 시도해 보지 못한채 경로가 단절되었다고 판단하게 된다. 그 후 라우팅 프로토콜에 의해 경로 재설정

상태로 들어가게 되는데 이는 TCP 성능 저하를 일으킨다. 따라서 노드가 최대 재전송을 시도하는데 걸리는 시간이 hidden terminal이 한 프레임을 전송 시간보다 커야 한다. 예를 들어 그림 1에서 4번 노드가 5번 노드로 1 Kbyte의 패킷을 전송 중에 있고 1번 노드가 2번 노드에게 전송 시도를 한다면 1번 노드가 전송시도를 하는 시간이 4번 노드가 패킷을 전송하는 시간에 비해 커야 한다. 4번 노드가 1 Kbyte의 패킷을 전송하는데 걸리는 시간은 그림 3을 통해 구할 수 있다. 대역폭을 2 Mbps로 사용한다고 하면 1 Kbyte의 패킷을 전송하는데 걸리는 시간은 4 ms가 소요된다. 따라서 4번 노드가 소요하는 총 시간은 다음 식(1)과 같이 계산된다.

$$\begin{aligned} Tx &= RTS+3*SIFS+CTS+DATA+ACK \\ &= (44byte\ 전송시간)+(3*10\mu s)+(38byte\ 전송시간) \\ &\quad +(1000byte\ 전송시간)+(38byte\ 전송시간) \\ &= 0.175ms+30\mu s+0.152ms+4ms+0.152ms \\ &= 4.479ms \end{aligned} \quad (1)$$

따라서 RTS를 7번 시도하는 시간이 4.479 ms보다 작을 경우에는 hidden terminal로 인하여 성능 저하가 발생된다. CW값에 따른 7번의 RTS를 시도하는데 소요되는 시간은 표 1과 같다. 이 시간은(0, $CW_{\min}$)내의 값을 선택해서 충돌 회피를 하며 7번 RTS를 시도하는데 걸리는 시간으로 $\frac{CW_{\min}}{2}*20\mu s*7+RTS\ timeout$ 이 된다.

표 1과 같이 $CW_{\min}=15$ 일 경우 7번째 RTS를 시도하는 시간이 hidden terminal이 1 Kbyte를 전송하는 시간보다 작기 때문에 제안된 기법을 사용하여 7번의 RTS를 시도해도 성능 향상을 가져올 수 없다. 따라서 본 기법에서 가질 수 있는 최소 크기의 CW는 $CW_{\min}=31$로 설정하고 $CW_{\min}=63$이상으로 설정해야 hidden terminal을 해결할 수 있는 조건을 갖게 된다.

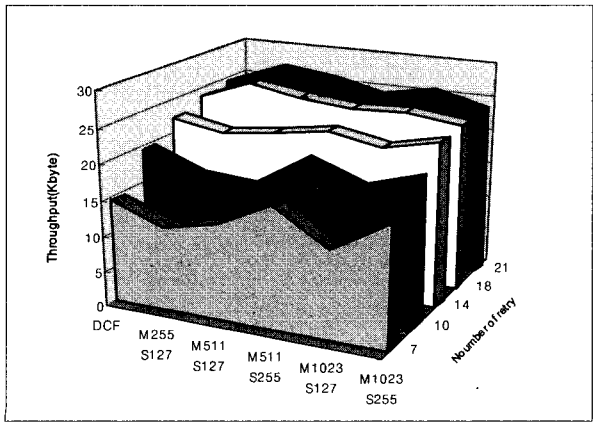


그림 4. Chain topology에서 Adaptive DCF의 throughput
Fig. 4. Throughput of Adaptive DCF in chain topology.

Ⅳ. 시뮬레이션을 통한 성능 분석

Adaptive DCF의 동작을 확인하기 위해서 먼저 그림 1과 같은 chain topology 환경에서 ns-2[8]를 사용하여 시뮬레이션을 수행하였다. Chain topology에서는 노드의 이동성을 고려하지 않았으며 8홉의 환경에서 일반 DCF를 Adaptive DCF와 비교해 보았다. Chain topology는 hidden terminal이 존재하지만 주변의 노드가 적고, 이동이 없기 때문에 그 영향이 작다고 볼 수 있다. 그림 4는 8홉의 chain topology망에서 제안한 Adaptive DCF의 파라미터를 변화시키면서 일반 DCF와 성능 비교를 한 그림이다. 시뮬레이션은 DSR[7] 라우팅 프로토콜을 사용하였고 500초 동안 시뮬레이션을 수행하였다. 대역폭은 2 Mbps로 설정하였다. 그림에서 Adaptive DCF의 parameter M은 $CW_{\max}$값을, S는 CW_{normal}값을 의미한다. 이 그림에서 보면 재전송 횟수(retry)를 늘리면 throughput이 개선됨을 알 수 있다.

재전송 횟수의 증가는 hidden terminal이나 일시적인 혼잡으로 인해 전송에 실패하였을 경우 경로 재설정을 제한하여 불필요한 경로 재설정을 수행하지 않음으로 얻는 성능 향상이다. Chain topology는 소스 노드에서 목적지 노드로의 경로가 일정하기 때문에 경로 재설정에 의한 성능 차이가 크지는 않지만 경로가 다양한 grid 환경에서는 경로 재설정의 영향이 크다. 재전송 횟수의 증가에 의한 경로 재설정의 영향은 grid 환경에서 살펴보겠다.

그림 4를 보면 두 parameter에 의한 성능의 차이가 존재하는데 전반적으로 CW_{normal}과 $CW_{\max}$의 값을

표 1. CW값에 따른 7번 RTS 재전송 시도 시간
Table 1. 7 RTS retransmission time for various CW.

$CW_{\min}$ 값	평균 CW의 backoff 시간값	RTS time out	7번 RTS 재전송 시간
$CW_{\min}=15$	$8*20\mu=160\mu s$	$0.478ms*6$	$3.988ms$
$CW_{\min}=31$	$16*20\mu=320\mu s$	$0.478ms*6$	$5.108ms$
$CW_{\min}=63$	$32*20\mu=640\mu s$	$0.478ms*6$	$7.348ms$

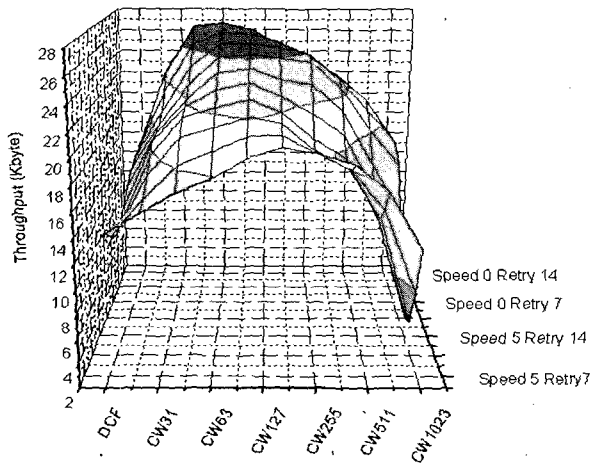


그림 5. Constant DCF에서 재전송 횟수와 이동성에 따른 TCP throughput 비교

Fig. 5. TCP throughput comparison of constant DCF for varying mobility and retry count.

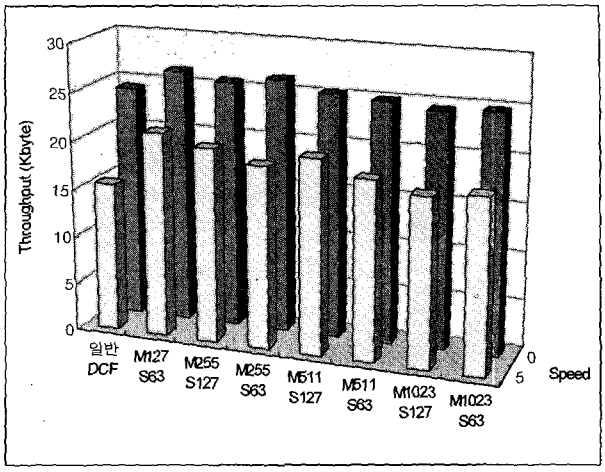


그림 6. 일반 DCF와 Adaptive DCF와의 성능 비교

Fig. 6. Throughput comparison between original DCF and Adaptive DCF.

작게 설정할 때 성능이 좋다. 이는 chain topology에서는 노드의 주변에 있는 노드의 수가 한정적이며 충돌로 인한 전송 실패의 가능성이 작기 때문에 일반 DCF에서 사용하는 $CW_{\max}$ 값인 1024보다 작은 값이 좋다. Chain topology에서는 노드간 충돌이 크게 발생되지 않기 때문에 일반 DCF와 큰 성능 차이는 보이지 않는다.

다음으로는 7 x 7 grid topology에 노드를 200m 간격으로 위치시키고 노드의 이동 속도를 변화시키며 시뮬레이션을 수행하였다. 노드의 이동 시나리오는 자신의 초기 위치를 중심으로 일정시간(5초)마다 random한 좌표를 받아 일정한 속도인 5m/s 또는 10m/s로 이동시켰다. 경로가 존재하지 않아 패킷이 전송되지 못하는 것을 방지하기 위해 노드의 이동 범위를 초기 위치에서 일정 범위(50~200m)내로 이동 범위를 제한함으로 경로가 없는 것을 방지하였다. 대역폭은 2 Mbps를 사용하였고 500초간 시뮬레이션을 수행하여 결과값을 산출하였다. 각 기법들의 성능을 시간당 throughput을 통하여 비교분석하였다.

그림 5는 DCF에서 충돌여부와 상관없이 CW를 고정하여 사용하는 Constant DCF 방법에서 노드의 이동 속도와 재전송 횟수에 따른 결과값을 나타내고 있다. 이 실험은 hidden terminal 문제가 exponential backoff에 의해서 더 증폭됨을 보여주기 위해 충돌여부와 상관없이 고정 CW를 사용한 것이다. 결과 그래프를 통해서 알 수 있듯이 일반 DCF보다 성능이 개선되며 CW의 값을 63이나 127과 같이 작은 값으로 설정하는 것이 성능이 우수함을 알 수 있다. 노드의 속도가 커짐에 따라 성능 저하가 발생되며 재전송 횟수가 많을수록 성능이 향상됨을 알 수 있다. 너무 작은 CW값인 31로 설정할 경우 노드간 충돌이 자주 발생하여 성능이 낮게 나오며 지나치게 큰 값인 1023으로 설정하면 노드의 이동이 없을 경우에는 충돌확률이 적어 재전송 횟수에 큰 영향을 받지 않는 반면 노드간 대기 시간이 길어져 성능이 낮게 나오며 노드의 이동이 있을 경우에는 경로 변경에 빠르게 반응하지 못해 성능이 감소된다. 노드가 이동할 경우 CW의 값을 255이상으로 설정할 경우 큰 CW값이 경로 변경에 빠르게 반응하지 못함으로 성능이 감소되며 큰 CW값을 가지고 재전송 시도를 높이면 오히려 성능이 감소됨을 알 수 있다.

그림 6에서는 제안한 Adaptive DCF의 파라미터값을 변경하며 일반 DCF와 비교 분석하였다. Grid topology의 환경에서 노드의 이동이 존재하지 않을 경우 경로의 변경과 노드간 충돌, hidden terminal의 영향이 적기 때문에 Adaptive DCF와 일반 DCF와의 성능의 차이가 크게 존재하지는 않지만 노드의 이동이 고려되는 경우 성능의 차이가 확연하게 나타남을 알 수 있다. 이는 노드가 이동을 하게 되면 경로 변경이 발생되고 또한 hidden terminal 문제가 심하게 발생되기 때문이다. 노드가 이동하여 경로가 변경되면 일반 DCF는 CW값을 증가시키며 재전송을 시도한 후에 경로 재설정을 수행하기 때문에 경로 변경에 적합한 반응을 하지 못해 성능이 저하된다. 반면 Adaptive DCF의 CW 파라미터를 작게 설정할 경우 노드의 이동에 빠르게 반응하며 hidden terminal로 인해 CW가 증가되어 전송대기 시간이 증가되는 것을 방지하기 때문에 노드의 이동으로

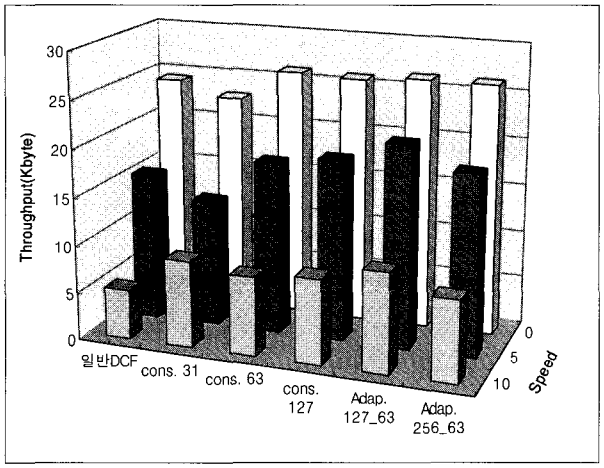


그림 7. 노드의 이동 속도에 따른 throughput 비교 (Retry 7)

Fig. 7. Throughput comparison for various node's speed with retry=7.

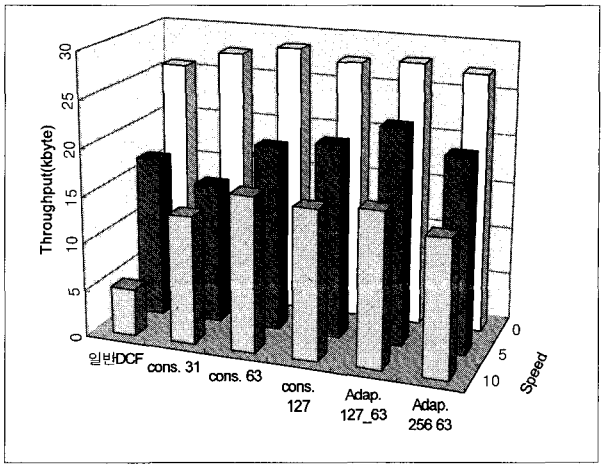


그림 8. 노드의 이동 속도에 따른 throughput 비교 (Retry 14)

Fig. 8. Throughput comparison for various node's speed with retry=14.

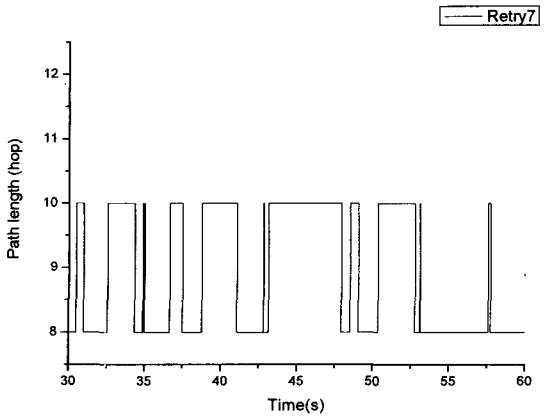
인해 성능 저하가 발생되지만 그 성능 저하가 일반 DCF보다 상대적으로 작게 발생되며 성능이 우수함을 알 수 있다.

그림 7은 일반 DCF와 Constant DCF, Adaptive DCF를 노드의 이동 속도에 따라 성능 분석을 한 결과 그래프이다. Constant DCF와 Adaptive DCF는 앞에서 수행한 시뮬레이션 결과를 통해 최적의 파라미터를 선출하였고 이를 가지고 시뮬레이션을 수행하여 성능 비교를 하였다. 노드가 이동하지 않을 경우 Constant DCF와 Adaptive DCF의 성능이 일반 DCF에 비해 약 8%의 성능 향상을 보인다. 하지만 노드가 이동할 경우 성능 향상이 크게 발생된다. 5m/s의 이동 속도에서는 Adaptive DCF가 약 35%의 성능 향상을 보이며 노드의 속도가 더 빠른 10m/s의 환경에서는 약 100%의 성능 향상을 보인다. Adaptive DCF는 작은 CW의 값을 갖는 Constant DCF와 비교하였을 때도 노드의 속도가 5m/s에서는 17%, 10m/s에서는 24%의 성능 향상을 보인다. 이와 같은 결과는 Adaptive DCF가 작은 CW의 값을 가짐으로 성능 향상을 보이지만 hidden terminal로 인해 발생되는 성능 저하를 줄임으로 Constant DCF에 비해 성능이 우수하게 나타남을 알 수 있다.

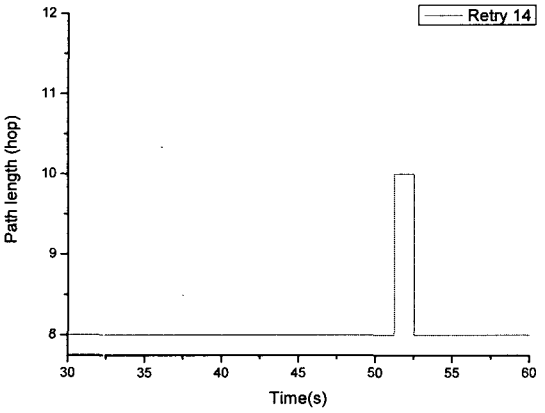
그림 8은 재전송 시도 횟수를 14로 증가하였을 때의 결과값을 비교한 것이다. 재전송 시도 횟수를 증가시키면 약간의 성능 향상이 존재하며 각 기법간 성능은 재전송 시도 횟수에 상관없이 일정한 특성을 나타난다. 앞서 살펴본 바와 같이 링크 계층에서 7번의 재시도가 실패하면 ad-hoc 라우팅 프로토콜은 경로가 끊어지거나 변경되었다고 판단하고 새로운 경로를 찾기 위해 경로 설정을 위한 라우팅 메시지를 전송하게 된다. 이 패킷은 경로를 찾을 때까지 브로드캐스트되어 새로운 경로를 찾게 된다. 하지만 ad-hoc 환경에서는 간섭이나 노드의 이동으로 인해 경로가 일시적으로 끊어진 것처럼 보일 때가 있다. 이와 같은 경우 라우팅 메시지가 전송되는 것은 불필요한 절차이며 이 때 전송되는 라우팅 패킷이 망 대역폭을 낭비하는 문제가 발생될 수 있다. 따라서 무선망의 특성을 고려하여 재전송 횟수를 증가시킬 경우 불필요한 경로 재설정 절차를 제한함으로써 성능 향상을 얻을 수 있다.

그림 9는 grid topology에서 Adaptive DCF의 parameter를 M 127, S 63으로 설정하였을 때 재전송 횟수에 따른 경로 변화를 나타내고 있다. 노드의 이동을 고려하지 않았을 경우 일시적인 간섭이나 망 혼잡으로 인해 일시적으로 전송 실패가 발생될 수 있으며 이러한 경우 경로를 재설정하는 것은 오버헤드가 될 수 있다. 그림 9의 (a)와 (b)를 보면 알 수 있듯이 재전송 횟수를 7로 설정할 경우 30초 동안 20번 정도의 경로 재설정이 수행되는 반면 14로 설정할 경우 2번의 경로 재설정이 수행된다. 또한 최적의 경로인 8홉의 경로를 사용하는 시간의 차이가 뚜렷하게 존재한다. 따라서 재전송 횟수를 증가시키면 실제적인 경로 단절이 아닌 일시적인 전송 실패로 인한 경로 재설정을 제한하기 때문에 성능 향상을 얻을 수 있다. 이 그림을 통해 MAC 계층의 재전송 횟수에 따른 성능 차이의 원인을 알 수 있다.

표 2는 제안한 Adaptive DCF와 일반 DCF, Constant DCF를 최대 재전송 횟수를 7로 설정하고 비교하여 Adaptive DCF의 성능 향상을 나타내었다. 시뮬레이션



(a) 재전송 횟수가 7일 때



(b) 재전송 횟수가 14일 때

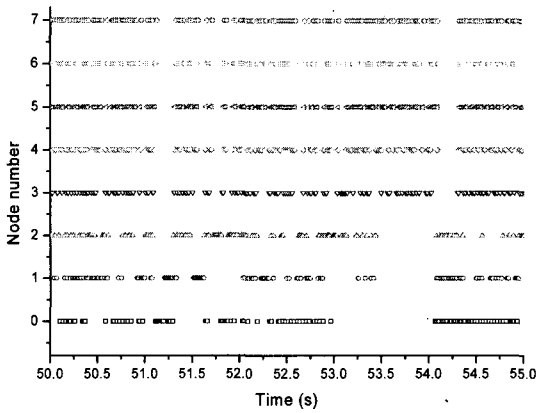
그림 9. 재전송 횟수에 따른 경로 길이 변동
Fig. 9. Path length variation for the number of retry.

표 2. Adaptive DCF의 성능 향상
Table 2. Throughput improvement of Adaptive DCF.

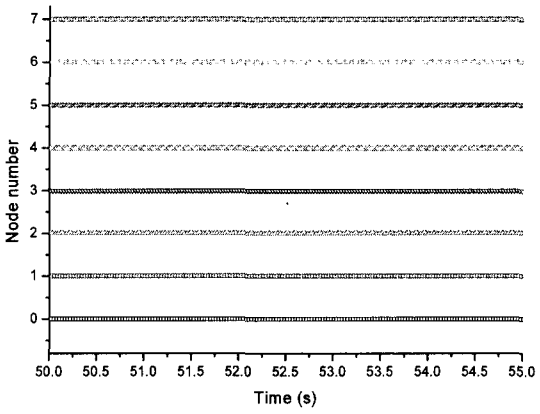
[단위 Kbyte]

노드의 속도에 따른 성능		일반 DCF	Constant DCF	Adaptive DCF
0 m/s	throughput	24.6	26.1	26.4
	Adaptive DCF의 성능향상률	8.6%	1.1%	N/A
5 m/s	throughput	15.5	18.05	21.25
	Adaptive DCF의 성능향상률	37%	17.7%	N/A
10 m/s	throughput	5.2	8.21	10.46
	Adaptive DCF의 성능향상률	101.1 %	24.4%	N/A

결과를 통해 알 수 있듯이 이동 ad-hoc 망에서는 노드가 이동함으로 발생되는 경로 변경에 빠르게 반응하기



(a) 일반 DCF



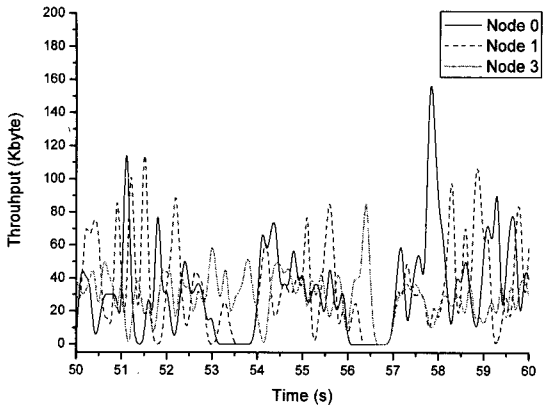
(b) Adaptive DCF

그림 10. 노드간 전송 기회의 공평성 비교
Fig. 10. Fairness of transmit between nodes.

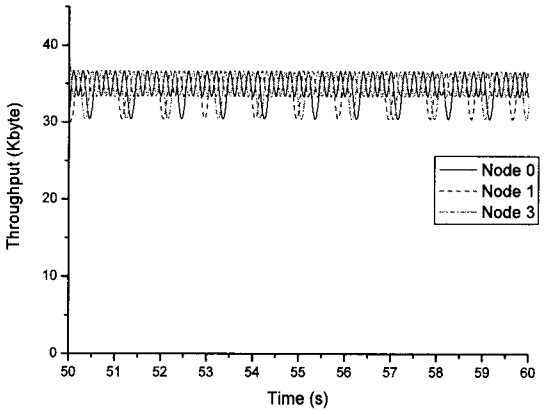
위해서는 작은 CW의 값을 갖는 것이 효율적이며 노드가 전송을 시도하다가 충돌이 발생하여 패킷 손실이 발생되었을 때 다시 재전송을 하기 위해 random backoff 과정을 거치게 되는데 이 때 충돌의 원인을 고려하여 hidden terminal일 경우는 CW를 증가시키지 않고 반으로 줄이고 정상적인 상태에서는 CW를 유지시키고 망 혼잡 상태에서만 CW를 증가시키는 방식을 사용함으로 성능 향상을 얻을 수 있다.

그림 10과 그림 11은 일반 DCF와 Adaptive DCF의 노드간 전송 기회의 공평성을 비교 분석하고 있다. 노드 9개를 그림 1과 같이 chain topology에 배치시키고 hidden terminal의 영향과 전송기회의 공평성을 분석하기 위해 대역폭 2 Mbps, 패킷 사이즈 1 Kbyte, 패킷간 간격을 30 ms로 설정한 UDP 트래픽을 8홉 떨어진 0번 노드와 8번 노드 사이에 전송하였다. 이때 Adaptive DCF의 파라미터는 M 128, S 63으로 설정하였다.

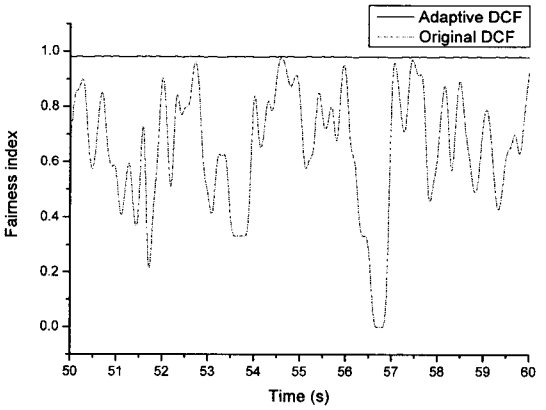
그림 10에서는 각 노드가 전송 기회를 얻어 패킷을



(a) 일반 DCF



(b) Adaptive DCF



(c) Fairness index

그림 11. Adaptive DCF와 일반 DCF의 공평성 비교
Fig. 11. Fairness comparison between the original DCF and the Adaptive DCF.

전송한 시간을 나타내었다. (a)에서 보듯이 일반 DCF의 경우 노드의 전송에 있어 burst한 특성을 나타내는데 이는 hidden terminal(0일 경우 3)로 인해 노드가 전송은 하지 못하고 CW만 증가하여 전송대기 시간이 증가하고 전송에 성공한 노드는 계속해서 전송 기회를 얻기 위한 경쟁에 있어 우위를 차지하기 때문이다. 예를 들어 53초에서 54초 사이에 0번 노드는 채널이 유휴한 상태라고 판단하지만 hidden terminal인 3번 노드가 전송 중에 있기 때문에 전송을 하지 못하게 되고 최대 재전송 회수만큼 시도 후 경로가 손실되었다고 판단하고 경로 재설정을 수행한다. 반면 (b)에서 보듯이Adaptive DCF는 작은 CW를 설정하여 노드간 전송대기 시간이 작아 망 사용 효율성이 높으며 hidden terminal로 인해 전송 실패할 경우에는 CW를 증가시키지 않고 감소시키기 때문에 노드간 전송 기회의 공평성을 보장한다.

그림 11에서는 각 노드들의 시간당 throughput을 측정하였다. (a)에서와 같이 일반 DCF의 경우 노드가 전송 기회를 얻었을 때 burst한 트래픽이 생성되기 때문에 노드별 시간당 throughput의 차이가 크게 발생된다. 반면 (b)에서와 같이 Adaptive DCF의 경우 전송 기회가 비교적 공평하게 주어지며 작은 CW 파라미터를 사용함으로 전송대기 시간이 짧아 각 노드별 throughput이 비슷하게 나타나고 일반 DCF와 비교하였을 때 평균 throughput이 우수하다. (c)에서는 0번 노드와 1번 노드 3번 노드의 공평성을 다음과 같은 식(2)로 표현되는 Jain's fairness index[9]를 통해 나타내었다.

$$\text{Fairness index} = \frac{(\sum x_i)^2}{n \times \sum (x_i)^2} \tag{2}$$

식(2)에서 n은 공평성 고려 대상의 수를 나타낸다. 공평성 지수가 1.0일 경우 100% 공평함을 의미한다.

V. 결론 및 향후 연구 계획

본 논문에서는 이동 ad-hoc 망에서 TCP 성능에 가장 큰 영향을 미치는 hidden terminal 문제를 살펴보았고 hidden terminal의 영향을 최소화시키기 위한 방안을 제안하였다. Ad-hoc에서 사용되고 있는 802.11 MAC의 DCF에서 사용하는 RTS/CTS 방식은 hidden terminal 문제를 해결하지 못하며 random backoff 방식에서 전송 실패하였을 때 CW를 지수적으로 증가시키는 방법은 노드의 이동이 많은 이동 ad-hoc 망 환경에 적합하지 못하며 한 노드에 의해 burst한 트래픽이 생성되는 양상을 갖는다. 따라서 본 논문에서는 이동 ad-hoc 망 환경에 적합하도록 기존의 DCF 방식을 개선한 Adaptive DCF 방식을 제안하였다. 이 방식을 사용할 경우 전송 실패를 하였을 때 그 원인을 판단하여 그에 적합하도록 CW를 조절함으로 성능 향상을 얻을

수 있다. 지수적인 CW 증가를 할 경우 발생되는 전송 기회의 불공평성을 해결함으로 hidden terminal로 인해 노드의 전송기회가 줄어드는 것을 막음으로 노드 간 전송 기회의 공평성을 보장한다. 또한 노드가 이동하는 망에서 CW를 작은 값으로 설정함으로 망 변화에 빠르게 반응하여 손실된 경로를 빠르게 복구시키고 전송 대기 시간을 줄이며 재전송 횟수를 증가시킴으로 경로 재설정에 따른 오버헤드를 줄여 성능 향상을 얻을 수 있다.

향후 연구 계획으로는 다중 트래픽 흐름이 존재할 경우 Adaptive DCF의 성능을 최적화하기 위하여 NAV_{count} 값에 따른 적절한 CW 변화 함수를 찾고 노드의 이동 속도에 따른 재전송 횟수와의 상관관계를 연구하여 이동 ad-hoc 망 환경에서의 성능 향상을 이루고자 한다.

참 고 문 헌

[1] C. E. Perkins, *"AD HOC NETWORKING"*, Addison Wesley, 2001.

[2] E. M. Royer and C. Toh, "A review of current routing protocols for Ad-hoc mobile wireless networks", IEEE Personal Communication, pp. 207-218, April. 1999.

[3] K. Chandran, S. Raghunathan, S. Venkatesan, P. Prakash, "A Feedback-Based Scheme for Improving TCP Performance in Ad Hoc wireless Networks," IEEE Personal Communications, February 2001.

[4] The Editors of IEEE 802.11. IEEE Standard for Wireless LAN Medium Access Control(MAC) and Physical Layer(PHY) specifications, Nov. 1997.

[5] H. S. Chhaya and S. Gupta, "Performance modeling of asynchronous data transfer methods of IEEE 802.11 MAC protocol", Wireless Networks, vol. 3 (1997), pp. 217-234, 1997.

[6] T. S. Ho and K. C. Chen, "Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LAN's," in Proc. IEEE PIMRC, Taipei, Taiwan, pp. 392-296, Oct. 1996.

[7] D. B. Johnson, D. A. Maltz, Y. Hu, "Dynamic Source Routing Protocol for Mobile Ad Hoc Networks(DSR)", Internet Draft <draft-ietf-manet-dsr-09.txt>, April 2003.

[8] Network Simulator, ns version2-29, http://www.isi.edu/nsnam/ns/

[9] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer system," Digital Equipment Corporation, Technical Report, DEC-TR-301, Sep. 1984.

저 자 소 개

김 한 집(학생회원)
2004년 한남대학교 전자정보 통신공학과 학사
2004년~현재 충남대학교 정보통신공학과 석사과정
<주관심분야: 이동통신 네트워크, 데이터 통신, Mobile IP>

이 재 용(종신회원)-교신저자
1988년 서울대학교 전자공학과 학사
1990년 한국과학기술원 전기 및 전자공학과 석사
1995년 한국과학기술원 전기 및 전자공학과 박사
1990년~1995년 디지콤 정보통신 연구소 선임연구원
1995년~현재 충남대학교 정보통신공학부 부교수
<주관심분야 : 초고속통신, 인터넷, 네트워크 성능분석>

이 기 라(학생회원)
2005년 배재대학교 정보통신공학부 학사
2006년~현재 충남대학교 정보통신공학과 석사과정
<주관심 분야 : 이동인터넷, 이동통신 네트워크, 데이터통신>

김 병 철(종신회원)
1988년 서울대학교 전자공학과 학사
1990년 한국과학기술원 전기 및 전자공학과 석사
1996년 한국과학기술원 전기 및 전자공학과 박사
1993년~1999년 삼성전자 CDMA 개발팀
1999년~현재 충남대학교 정보통신공학부 부교수
<주관심분야 : 이동인터넷, 이동통신 네트워크, 데이터통신>