Ad-hoc 무선 망에서 DSR 라우팅 프로토콜을 이용한 인터넷 트래픽의 성능 분석

이규남°, 고영웅, 육동철, 박승섭
부경대학교 교육대학원 전자계산학과

Performance Analysis of TCP Traffic over DSR Routing Protocol in Ad-Hoc Wireless Network

Kyu-Nam Lee°, Young-Woong Ko, Dong-Chul Yuk, Seung-Seob Park
Dept. of Computer Science, Graduate School of Education, PuKyong National University

요 약

Ad-hoc 무선망은 고정된 유선망을 가지지 않고 이동 노드들로만 구성된 망이다. 여러 가지 이동 노드에 대한 이동 제약이 없고, 유선망이나 기지국 같은 기반 구조를 필요를 하지 않기 때문에 여러 환경에 적용이 가능하다. 그러나 라우터들 사이의 이동성과 다른 연결 요소들의 가변적인 요소는 잠재적으로 속도감 있고 예측 불가능하게 변하는 망을 가져올 수 있다. 이에 따른 패킷의 손실이 무선 네트워크에서 자주 발생하게 된다.

최근 몇 년 동안 Ad-hoc망에서 사용되는 라우팅 프로토콜의 각각에 대한 성능을 분석하는 연구가 있었지만, Ad-hoc 무선망에서 특정한 프로토콜의 TCP 버전별 비교 분석에 대한 연구가 미비한 실정이다

따라서 본 논문에서는 무선 네트워크가 가지는 단점을 보완하고 트래픽 성능을 향상시키기 위한 한 방편으로, Ad-hoc 라우팅 알고리즘인 DSR 프로토콜을 이용하여 TCP Tahoe, Sack, Reno 버전별로 토폴로지의 크기의 변화와 이동 노드의 이동 속도의 변화에 따라 트래픽의 성능을 모의 실험하여 비교 분석하였다.

1. 서론

Ad-hoc 무선망은 병원, 전시장, 생산 현장 등과 같은 긴박한 상황이나 지속적인 망 연결이 필요 없는 환경에서 적용 가능하다. Ad-hoc 무선 네트워크에서 각각의 이동 노드는 단지 호스트가 아니라 하나의 라우터로 동작하며, 다른 노드에 대해 다중 경로를 가질 수 있다. 또한 동적으로 경로를 설정할 수 있기 때문에 기반구조 없는 네트워킹이라고도 한다.

특정 시점에서 이동노드의 통신 연결 상태는 노드 사이의 위치나 전송 전력레벨, 안테나 패턴, 채널간의 상호 레벨의 함수로 나타낸다. 라우터들 사이의 이동성과 다른 연결 요소들의 가변적인 요소는 잠재적으로 속도감 있고 예측 불가능하게 변하는 망을 가져올 수 있다. 이에 따른 패킷의 손실이 무선 네트워크에서 자주 발생하게 된다[1].

TCP는 널리 알려져 있고, 가장 많이 사용되는 인터넷 프로토콜이다. TCP를 이용하여 여러 가지 어플리케이션을 사용 할 수 있도록 하기 위해 Ad-hoc 무선망에서도 TCP를 사용하여야 한다. 그러나 Ad-hoc 무선망은 기존의 일반 유선 망과 달리 end-to-end 전달 방식을 사용하는 것이 아니라 홉 단위로 데이터를 전달하는 hop-by-hop 전달 방식을 사용하므로 신뢰성 있는 서비스를 제공하지 못한다. 기존의 연구에서는 실시간 처리를 하는 특정한 하나의 인터넷 트래픽에 관한 연구나 Ad-hoc 무선망에서 사용되는 라우팅 프로토콜의 각각에 대한 성능을 분석하는 연구가 있어 왔다[2][3]. 또한 인터넷 트래픽을 시뮬레이션 파라미터로 사용하여 라우팅 프로토콜의 성능을 분석한 연구가 있었지만, Ad-hoc 무선망에서 특정한 프로토콜의 TCP 버전별 비교 분석에 대한 연구가 미비한 실정이다[4]. 따라서, 본 논문에서는 Ad-hoc무선 네트워크에서 대표적인 On-Demand 라우팅 프로토콜인

DSR을 사용하였다. 이 On-Demand 라우팅 프로토콜의 성능을 평가하기 위해 TCP Reno버전과 Sack 버전, Tahoe 버전의 트래픽 성능을 비교 분석하였다. 본 모의 실험에서는 이동 노드의 이동 속도 변화와 토폴로지의 크기를 변화하는 방식으로 실험을 하였다.

본 논문의 구성은 1장 서론에 이어, 2장 관련연구에서는 Ad-hoc 무선망에서 사용되는 On- Demand 라우팅 프로토콜인 DSR의 메커니즘과 TCP의 버전별 특징에 대해서 설명한다. 3장에서는 본 연구에서 사용된 NS-2 시뮬레이터와 시뮬레이션 환경에 대해 설명을 하고 있으며, 4장에서는 모의 실험 결과 분석에 대해서 설명하고, 마지막 5장에서 본 논문의 결론에 대해 서술하였다.

2. 관련연구

본 장에서는 DSR 프로토콜의 개념 및 알고리즘에 대한 분석, 그림고 TCP 제어 알고리즘에 대해 설명한다.

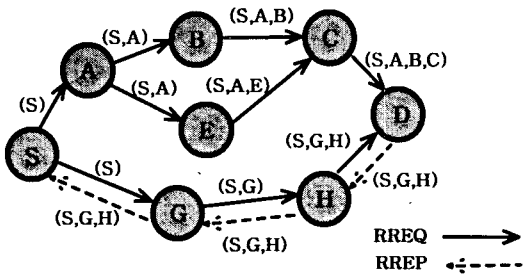
2.1 DSR(Dynamic Source Routing) 프로토콜

DSR은 카네기멜론 대학의 Monarch 프로젝트('96 Johnson)에 의해 제안된 On-Demand 방식의 대표적인 프로토콜로서 기본적으로 소스 라우팅의 알고리즘을 사용한다[4][5][6]. 소스 라우팅을 DSR에서 사용함으로써 출발지 이동 단말은 경로가 필요할 때 경로 획득절차를 수행하여 얻은 경로를 데이터 헤더에 추가하여 전송한다. 소스 노드가 목적지 노드로의 경로를 모두 알고 있어 소스에서 목적지로 전송되는 모든 데이터는 헤드에 그 경로를 포함하고 있다. 따라서 복잡한 경로를 가진 경우에는 전송되는 패킷의 데이터보다 헤드가 더 많을 수도 있다. 이 알고리즘을 경로 탐색 단계(Route Discovery)와 경로 유지 단계(Route Maintenance)로 나누어 분석한다.

2.1.1 DSR의 경로 탐색 단계

DSR 알고리즘에서는 어떤 소스 노드가 다른 목적지 노드로 패킷을 보내려고 할 때 모든 라우터들에 대한 정보를 알고 있지 않은 관계로 소스 노드에서 목적지 노드로의 경로에 대한 정보를 알 수 있어야 한다. 이 때 (그림 1)과 같이 경로탐색 동작이 일어난다.

목적지 노드에 대한 경로 탐색 단계는 소스 노드가 먼저 라우트 캐쉬에서 목적지 노드에 대한 정보를 검색한다. 이 때 목적지 노드에 대한 정보가 없을 경우 소스 노드는 경로 탐색 단계를 수행하게 되는데 소스 노드는 경로 요구(RREQ: Route Request) 메시지를 망 전체로 전달함으로써 시작한다.



(그림 1) 경로 탐색 동작

메시지 전달 이후 소스 노드 주위의 모든 노드들은 그 패킷을 받게 되고 각각의 라우터들은 자신이 가지고 있는 라우트 캐쉬의 경로를 조사한다. 그리하여 RREQ패킷의 목적지 노드의 값이 그 노드의 라우트 캐쉬에 있게 되면 그 노드는 RREP 패킷을 반환하고 그렇지 않을 경우에는 수신한 RREQ 패킷을 다른 이웃 노드들에게 재 전송한다. 이러한 방법을 통해 RREQ 패킷은 목적지 노드를 찾을 때까지 전 노드에 전송하게 된다.

소스 노드가 방송한 패킷을 중간에 있는 노드 A가 받았다면, 그 노드는 RREQ를 재전송하기 전에 먼저 자신의 라우트 캐쉬를 살펴보게 된다. 이 때, A노드의 라우트 캐쉬에 목적지 노드에 해당하는 경로가 저장되어 있으면 즉시 RREP를 생성하여 소스 노드에 보내주게 된다.

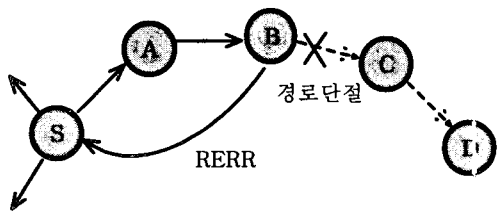
만일 노드 A의 라우트 캐쉬에 목적지 D로의 경로가 없을 때에는 소스 노드로부터 도착한 RREQ 패킷의 헤더에 현재 노드 A의 주소를 더하여 노드 A는 RREQ 패킷을 재 전송한다.

이러한 작업을 거친 후에 최종적으로 목적지 노드에 패킷이 도착하면 목적지 노드는 소스 노드에게 역경로로 RREP 패킷을 보내고 RREP 메시지에 의해 기억된 경로는 차후 데이터 전달에 사용되기 위해 소스 노드에 저장되고, (그림 1)의 망에서는 전달 할 데이터는 소스 노드(S)에서 G, H 노드를 거쳐 목적지 노드(D)에 전달하게 된다.

그런데 소스 노드에서 발생한 RREQ 패킷은 모든 노드에 방송되는 것이므로 같은 목적지로 두 가지 이상의 경로가 들어갈 수가 있게 되고, 목적지 노드에는 소스와 목적지는 같지만 경로가 다른 RREQ 패킷들이 각기 다른 경로를 통해 여러 개 들어올 수 있다. 이 때 목적지 노드는 먼저 받은 RREQ 패킷을 저장하고 있다가 다음 RREQ 패킷에서 소스와 목적지가 같으나 경로가 다른 것이 발견되면, 이후의 것을 모두 폐기한다. 왜냐하면 가장 먼저 도착한 것이 가장 빠른 경로이기 때문이다. 그렇기 때문에 소스 노드는 목적지 노드로부터 단 한 개의 RREP 패킷만을 받게 된다.

2.1.2 DSR의 경로 유지 단계

경로 유지 단계는 앞에서 획득한 경로를 보관/유지하는 알고리즘이다. 소스 경로 상에 있는 어떤 연결이 실패하여 목적지 노드까지의 경로가 여러 가지 이유로 더 이상 사용하지 못하게 되었을 때, 소스 노드는 경로 에러(RERR: Route Error) 패킷을 발생시켜 더 이상 그 경로를 사용하지 못하도록 하고 자신의 캐쉬로부터 실패한 경로를 제거하게 된다. 만일 소스 노드가 자신의 라우트 캐쉬에 다른 우회 경로가 존재하면 이 경로를 이용하여 데이터를 전송하고, 이러한 경로가 없을 때에는 다시 경로 탐색 단계를 시작하게 된다.



(그림 2) 경로 에러 전달 동작

(그림 2)에서처럼 소스 노드(S)에서 목적지 노드(D)로 패킷을 전송할 때 패킷 자체에는 A,B,C 노드를 거쳐서 전송되는 경로에 대한 정보가 포함되어있다. 최초 노드 S에서 노드 A로 패킷이 전달되면 노드 A에서는 노드 S로 응답신호를 보낸다. 이와 마찬가지로 B가 패킷을 받았을 때 B는 A에게 응답신호를 보낸다.

그런데 (그림 2)에서와 같이 S에서 A, B, C를 거쳐 D로 설정되어 있던 경로가 파손되었을 경우 노드 C는 노드 B에게 응답신호를 보내지 못하게 되고 노드 B가 그것을 감지하면 경로 에러에 대한 전달 동작이 일어나게 된다. 노드 B가 노드 C로부터 응답신호를 받지 못하거나 제한시간이 지나서 패킷을 재 전송하여야 할 경우에 노드 B는 목적지 노드 D로 가는 다른 경로를 라우트 캐쉬에서 검색한다. 만일 노드 B에 노드 D로 가는 다른 경로가 있으면 노드는 전달해야 할 데이터 패킷의 헤더를 지우고 라우트 캐쉬에서 검색된 새로운 경로를 패킷의 헤더로 바꾼다. 그러나 노드 B가 노드 D로 가는 다른 경로를 라우트 캐쉬에 가지고 있지 않으면 노드 B는 데이터 패킷을 버린다. 또한 노드 S로의 RREP도 생성하지 않는다. 대신에 노드 B는 소스 노드 S로 경로 에러 메시지를 보낸다. 소스 노드 S가 노드 B로부터 경로 에러 메시지를 받으면 S는 자신의 라우트 캐쉬에 저장되어있는 노드 D로 가는 경로를 지우고 경로 에러 메시지를 이웃노드들에게 전파하여 패킷이 전달되지 않았음을 알린다.

이와 같이 DSR은 Source Routing에 기반한 On-Demand 프로토콜로서 일정한 알고리즘으로 패킷이 전해지는 경로를 소스 노드가 모두 알아서 이후에 전해지는 데이터의 헤더에 그것들을 모두 포함시키는 방식을 취하고 있다. DSR은 패킷에 그 경로가 모두 있기 때문에 Table-driven 방식의 알고리즘보다 빠른 데이터 전송을 할 수 있다. 그러나 처음 경로를 찾을 때와 경로를 재 설정하는데 시간이 걸린다는 단점이 있다.

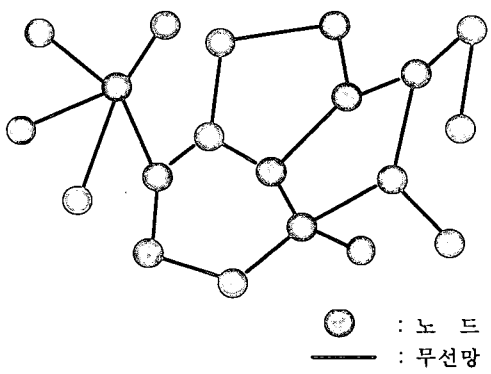
2.2 TCP 제어 알고리즘

Tahoe 버전은 초기에 Slow Start 기법에 따라 패킷을 송수신 하다가 혼잡이 발생하여 재전송 타임아웃이 발생하거나 중복응답이 수신되면 혼잡현상을 감지하여 저속출발 경계 값에 혼잡 윈도우 크기의 절반을 대입하여 다시 Slow Start를 실행한다. Reno 버전은 기존의 Slow Start, 혼잡회피(Congestion Avoidance) 알고리즘 이외에 빠른 회복(Fast Recovery)과 빠른 재전송(Fast Retransmission) 알고리즘을 추가한 버전이다. 세 개의 이중 긍정 응답이 수신되면 빠른 재전송을 수행하는 것은 Tahoe와 같으나 Slow Start 절차를 수행하는 대신 윈도우의 크기를 반으로 줄이고 재 전송한다.

Sack 버전은 개별적인 패킷에 대한 오류를 각각의 송신측에 피드백하여 오류제어의 성능을 높이는 방식으로 손실된 데이터 세그먼트 손실에 대한 응답을 두 개만 수신하면 재전송하기 때문에 Reno 버전보다 빠른 재전송이 가능한 것으로 알려져 있다[7].

3. 시뮬레이션 환경

본 장에서는 (그림 3)과 같이 모의실험에서 사용된 무선망의 모델과 파라미터에 대하여 설명한다. 시뮬레이션 툴로서는 버클리 대학에서 개발한 분산 객체 네트워크 시뮬레이터인 NS-2(Network Simulator)를 사용하였다[8].



(그림 3) 모의실험에서 사용된 무선망의 예

본 연구에서 라우팅 프로토콜을 분석하기 위해 비실시간 트래픽인 TCP 중에서 TCP Tahoe, Reno, Sack버전을 사용하였으며, 어플리케이션으로는 FTP를 사용하였다[9].

Ad-hoc 무선망의 라우팅 프로토콜은 대표적인 On-Demand 라우팅 프로토콜인 DSR을 사용하였고, TCP Tahoe버전, Sack버전, Reno 버전을 사용하여 각 버전별 성능을 비교 분석하였다.

본 실험에서는 아래 (표 1)과 같이 DSR의 TCP 버전별 처리율 관계를 분석하기 위해 이동 노드의 수 20개를 노드의 이동속도 10, 20, 30(m/s)로 가변 시키고 토폴로지의 크기도 300×300(m), 500×500(m), 800×800(m)로 변하게 하였다.

(표 1) 시뮬레이션에 사용 된 파라미터

토폴로지의 크기 (m×m)	300×300	500×500	800×800
이동 속도(m/sec)	10	20	30
트래픽 소스	TCP Tahoe, Reno, Sack		
데이터 패킷 크기	512 Byte		

각각의 모의실험에서 TCP 데이터 패킷의 크기를 512 Bytes로 고정하였고, TCP 데이터 패킷을 사용한 모의 실험 시간은 100초 동안 실험하였다. 모의 실험에서는 이동 노드의 유휴 시간은 0초로 설정하여 가변적인 크기의 토폴로지에 지속적으로 노드가 이동하는 망을 설계하였다.

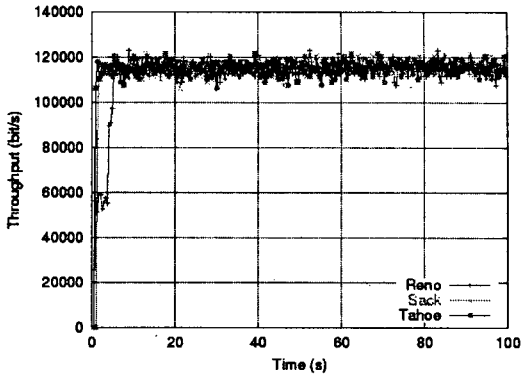
4. 시뮬레이션 결과 분석

본 장에서는 TCP 트래픽의 성능 분석 요소로서 패킷 처리율의 진동을 기준으로 하였고, 망의 지연율은 고려하지 않았다. 패킷의 처리율은 단위 시간당 목적지 노드가 수신한 패킷의 양에 시간을 나누어 처리율을 계산하였다.

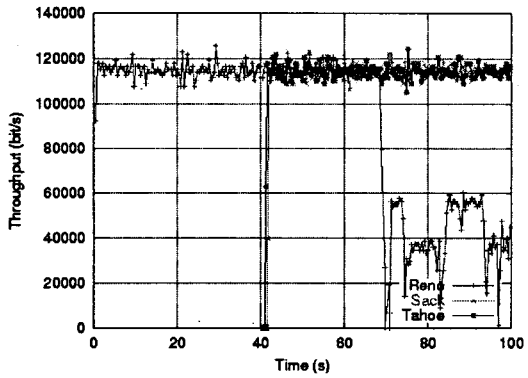
(그림 4)와 (그림 5)는 망의 크기 변화에 따른 DSR에서의 버전별 TCP 처리율의 변화를 나타낸 그림이다.

이동 노드의 속도를 고정하고 토폴로지의 크기를 변화시켜 비교 분석하였다. (그림 4)와 같이 이동 노드의 움직임이 빠르지 않고 토폴로지의 크기가 크기 않을 경우에는 모두 120kbps의 안정적인 처리율을 나타내었다.

(그림 5)와 같이 이동 노드의 속도는 고정시킨 채 망의 규모를 크게 하자 Reno 버전에서는 설정된 경로를 따라 안정적으로 패킷이 전송되다가 68초경에 패킷의 분실이 일어났다. 그러나 빠른 회복 알고리즘에 의해 경로 재 설정 과정이 일어나는 것을 볼 수 있었고, Sack 버전과 Tahoe 버전에서는 Reno 버전보다 40초 가량 늦게 연결 설정이 일어났지만 안정적으로 패킷이 전송되는 것을 볼 수 있었다.



(그림 4) 이동노드의 이동 속도가 20m/sec이고 토폴로지의 크기가 300m×300m일 때 Reno, Sack, Tahoe버전의 처리율



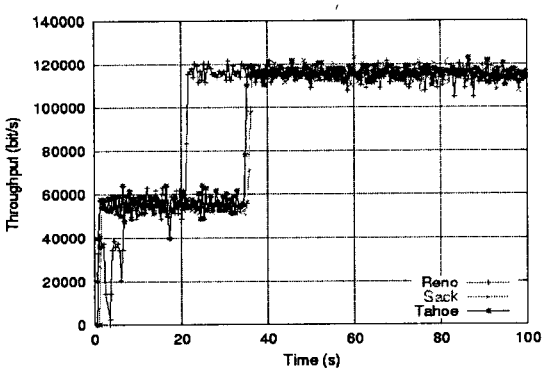
(그림 5) 이동노드의 이동 속도가 20m/sec이고 토폴로지의 크기가 800m×800m일 때 Reno, Sack, Tahoe버전의 처리율

(그림 6)과 (그림 7)은 이동 노드의 빠르기 변화에 따른 DSR에서의 버전별 TCP 처리율의 변화를 나타낸 그림이다.

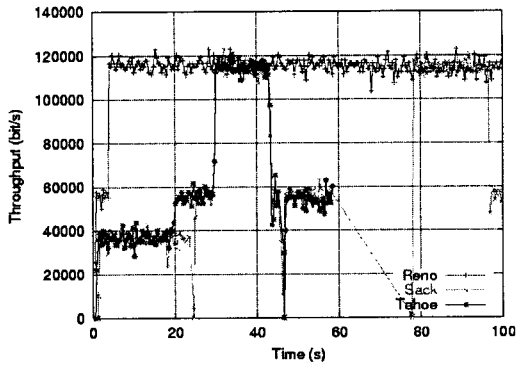
토폴로지의 크기는 고정하고 이동 노드의 이동 속도의 빠르기를 변화시켜 비교 분석하였다.

(그림 6)에서처럼 이동 노드의 이동 속도가 느릴 경우에는 경로설정시간이 Reno 버전에서 20초 정도, Sack 버전과 Tahoe 버전에서는 37초 정도 지나고 난 뒤 안정적으로 패킷이 전달되는 것을 볼 수 있었다. 그러나 (그림 7)에서처럼 이동 노드의 이동 속도가 빨라지자, Reno 버전에서는 여전히 안정적인 처리율을

나타냈으나, Sack 버전과 Tahoe 버전에서는 처리율의 변화가 심하게 발생함을 알 수 있었다.



(그림 6) 이동노드의 이동 속도가 10m/sec이고 토폴로지의 크기가 500m×500m일 때 Reno, Sack, Tahoe버전의 처리율



(그림 7) 이동노드의 이동 속도가 30m/sec이고 토폴로지의 크기가 500m×500m일 때 Reno, Sack, Tahoe버전의 처리율

5. 결론

최근 몇 년 동안 Ad-hoc망에서 다양한 라우팅 프로토콜이 연구되어 왔다. 그러나 이러한 프로토콜 상에서 인터넷 트래픽 성능에 관한 연구가 미진했다.

따라서 본 논문에서는 Ad-hoc 라우팅 알고리즘인 DSR 프로토콜의 전체적인 개념과 알고리즘 동작을 나타내었고, 망의 크기의 변화와 이동 노드의 이동 속도의 변화에 따른 트래픽의 성능을 비교하는 시뮬레이션을 하였다.

TCP Reno, Sack, Tahoe 버전에 따라 On-Demand 방식인 DSR을 이용하여 TCP 버전별 트래픽 성능을 비교 분석한 결과, 토폴로지의 크기가 그다지 크지 않을 때에는 Reno 버전의 경우 이동 노드의 속도에 관계없이 안정적으로 패킷을 처리하는 것을 볼 수 있었다. 그러나 토폴로지의 크기가 커지자 일정 시간이 흐른 뒤 패킷의 분실이 일어나고 다시 연결 재 설정하는 것을 볼 수 있었다.

Sack버전과 Tahoe 버전에서는 이동 노드의 이동 속도가 느린 경우에는 안정적으로 패킷을 전달하였으나, 이동 노드의 이동 속도가 빨라지자 처리율의 변화가 심하게 일어나는 것을 볼 수 있었다.

[참고문헌]

[1] S .Corson and J. Macker, "Mobile Ad-hoc Networking(MANET)", Internet Dreft, IETF, Oct, 1998.

[2] Charles E. Perkins, Elizabeth M. Royer and Samir Das, "Ad-Hoc On-Demand Distance Vector (AODV)Routing," Oct, 99 IETF Draft, pp, 33~38.

[3] J. Broch, D. A. Maltz, D. B. Johonson, Y, -C. Hu, and J. Jetcheva, "A performance Comparison of Multi-hop wireless Ad-Hoc Network Routing Protocols", In proceeding of the Fourth Annual ACM/IEEE international Conference on Mobile Computing and Networking, Oct. 1998.

[4] Josh Broch, D. A. Maltz. "The dynamic source routing protocol for Ad-hoc networks", draft-ietf-manet-dsr-01, Dec 1998.

[5] 이규남, 고영웅, 박승섭, "AODV와 DSR 라우팅 프로토콜 상에서 인터넷 트래픽의 성능비교", 한국 멀티미디어학회 춘계학술대회 논문집, 4권 1호, pp. 261~264, 2001년 6월

[6] D. Johnson and D. Maltz. "Dynamic source routing in Ad-hoc wireless networks.", Mobicom computing, 1996.

[7] K. Fall and S. Floyd, "Comparisons of Tahoe, Reno, and Sack TCP". ftp://ftp.ee.lbl.gov/papers/sack.ps.Z, December 1995.

[8] The VINT Project," ns Notes and Documentation", February 25, 2000.

[9] K. Fall and S. Floyd, "Comparisons of Tahoe, Reno, and Sack TCP". ftp://ftp.ee.lbl.gov/papers/sack.ps.Z, December 1995.