DSR 라우팅 프로토콜을 사용한 Ad-hoc 무선망에서의 TCP 성능 분석

박 승 섭† · 육 동 철††

요 약

Ad-hoc 무선망은 통신을 지원하는 베이스 스테이션과 같은 기반 구조를 가지지 않는 이동 노드들로만 구성된 망이다. 그러므로 노드의 이동으로 인해 Ad-hoc 무선망의 토폴로지가 자주 변하며, 이에 따른 패킷의 손실이 발생한다. 본 논문에서는 대표적인 On-Demand Ad-hoc 라우팅 알고리즘인 DSR 프로토콜을 사용하는 Ad-hoc 무선망에 TCP Tahoe, Sack, Reno 버전을 각각 적용하여, 망의 크기의 변화와 이동 노드의 속도의 변화에 따라 트래픽의 성능을 모의 실험하여 비교 분석하였다. 모의 실험 결과로, TCP Reno가 TCP Tahoe와 Sack 버전 보다 높은 처리율을 보였으며, 노드의 이동 속도와 망의 크기에 민감하지 않으므로 성능이 상대적으로 안정적이라는 것을 알 수 있었다.

Performance Analysis of TCP using DSR Routing Protocols in Ad-hoc Mobile Network

Seung-Seob Park† · Dong-Cheol Yuk††

ABSTRACT

Ad-hoc networks consist of a set of mobile hosts that communicate using wireless links, without the use of other supporting communication facilities (such as base stations, etc.). Therefore, the topology of an Ad-hoc network freguently changes due to the movement of mobile host, which may lead to sudden packet loss. Recently, the large amount of research has focused on the routing protocols needed in such an environment. In this paper, TCP Reno, Sack, and Tahoe versions are analysed using DSR protocol which is the representative On-Demand routing protocol in Ad-hoc wireless network. As the result of this simulation, we know that TCP Reno relatively has higher throughput than that of Sack and Tahoe, and TCP Reno has more stable performance than that of TCP Tahoe and Sack, regardless of the speed of mobile node and the size of topology.

키워드 : Ad-hoc 무선망(Ad-hoc wireless network), DSR 라우팅 프로토콜(DSR Routing Protocol), 트래픽 성능(Traffic Performance)

1. 서 론

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 이동 노드의 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 무선망에서는 기존 TCP의 종단간 전송 방식과 달리, 각각의 노드가 이동성을 가지므로 홉 단위로 데이터를 전달하는 방식을 사용한다. 따라서 경로 연결 실패가 잠재되어 있기 때문에 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]. 또한 인터넷 트래픽을 시뮬레이션 파라미터로 사용하여 라우팅 프로토콜의 성능을 분석한 연구가 있었지

※ 본 연구는 BK 지원금으로 수행되었음.
† 정 회 원 : 국립 부경대학교 전자컴퓨터정보통신공학부 교수
†† 준 회 원 : 부경대학교 대학원 전자계산학과
논문접수 : 2002년 7월 25일, 심사완료 : 2002년 9월 24일

만, Ad-hoc 무선망에서 특정한 프로토콜에서 TCP 버전에 대한 비교 분석과 TCP 버전 중에서 어떤 버전이 더 효율적인 전송을 하는지에 대한 연구가 미비한 실정이다[4].

따라서, 본 논문에서는 Ad-hoc무선 네트워크의 대표적인 On-Demand 라우팅 프로토콜중 하나인 DSR을 사용하고, 이동 노드의 속도와 무선망의 크기를 변화하여 TCP Tahoe 버전과 Reno 버전, Sack 버전의 트래픽 성능을 비교 분석하였다.

본 논문의 구성은 1장 서론에 이어, 2장 관련연구에서는 모의 실험에 사용된 TCP의 버전별 특징과Ad-hoc 무선망에서 사용되는 라우팅 프로토콜과 대표적인 On-Demand 라우팅 프로토콜인 DSR의 메커니즘에 대해서 설명한다. 그리고 3장에서는 시뮬레이션 환경에 대해 설명을 하고 있으며, 4장에서는 모의 실험 결과 분석에 대해서 설명하고, 마지막 5장은 본 논문의 결론이다.

2. 관련 연구

본 장에서는 TCP의 버전별 특징, Ad-hoc 무선 네트워크의 기본 개념과 특징을 설명하고, Ad-hoc 네트워크에 사용되는 라우팅 프로토콜과 DSR 알고리즘에 대해서 설명한다.

2.1 TCP

TCP는 인터넷에서 종단 시스템간의 신뢰성을 보장하는 데이터 전송 알고리즘으로 인터넷 서비스의 성능에 많은 영향을 미친다. 현재 TCP에는 망의 혼잡 상황에 대한 대처방식에 따라 여러 가지 버전이 있으며, 가장 많이 사용되는 TCP 버전인 Tahoe, Reno, Sack 버전에 대해서 설명한다.

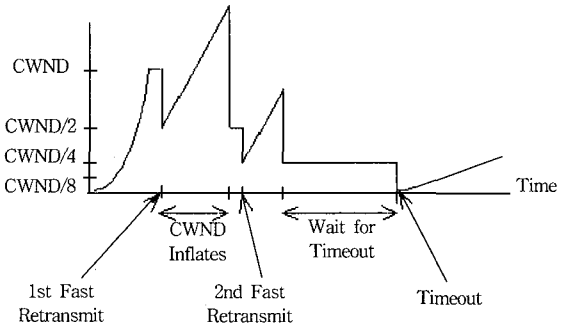
2.1.1 TCP Tahoe

TCP Tahoe 버전은 초기에 Slow-Start를 시작하여 패킷을 전달 하다가 혼잡이 발생하면 Slow-Start threshold(ss-thredhold)를 패킷이 손실되기 전의 반으로 재 설정한 후, 다시 Slow-Start를 실행한다. 만일 Congestion Window(cwnd) 값이 현재 설정된 ss-threshold값과 같아지면, 그 이후에는 혼잡회피 상태로 들어가게 된다.

Tahoe는 한 윈도우 내에 처음 발생한 손실된 패킷을 재전송한 후 차례로 Slow-Start를 실행하게 되고 이후의 모든 패킷을 재 전송하게 되므로 수신 호스트가 이미 수신한 패킷을 다시 수신하게 되는 문제점이 있으나 안정성이 높다[11, 13].

2.1.2 TCP Reno

TCP Reno는 기존의 Slow-Start, 혼잡회피 알고리즘 이외에 빠른 회복과 빠른 재전송 알고리즘을 추가한 버전이다. 빠른 재전송은 만일 수신 호스트에서 순서 번호에 어긋나는 세그먼트가 수신되면, 수신 호스트는 수신되지 않은 순서 번호로 확인 응답을 보내게 된다.



(그림 1) TCP Reno의 빠른 회복과 빠른 재전송

이 상황은 올바른 순서번호의 세그먼트를 수신할 때까지 계속된다. 따라서 송신 측이 중복된 확인 응답이 3개 이상 수신하게 되면 세그먼트가 손실되었다는 것으로 인식하여 세그먼트를 재전송 함으로서 타이머가 만료될 때까지 기다리는 시간을 줄이는 방법이다. 빠른 회복은 세그먼트 손실시 폭주 윈도우의 크기를 1로 설정하여 다시 Slow-Start 단계로 들어가는 방법을 개선하여 세그먼트 손실시 윈도우의 크기의 1/2을 현재 윈도우 크기로 설정하고, 바로 충돌회피 단계를 시작하여 Slow-Start 단계를 제거한다. 그러나 Reno 버전은 한, 두개의 세그먼트가 손실되었을 때는 성능이 우수하나 여러 개의 패킷이 손실되면 폭주 윈도우의 크기가 계속 줄어들어 일정 수 이상을 전달 할 수 없게 되는 상황이 발생할 수 있다[13]. (그림 1)는 TCP Reno의 빠른 회복과 빠른 재전송의 예를 나타낸다.

2.1.3 TCP Sack

Sack 버전은 Reno 버전의 세 개이상의 확인응답을 수신했을 때 재전송 하는 방식을 개선하여 선택적 확인응답을 사용하는 방식으로 동일한 확인응답이 두 개만 수신되어도 바로 세그먼트 손실로 인정하고 바로 재 전송하는 알고리즘을 추가한 것이다. Sack 버전은 세그먼트 손실에 대한 응답을 두 개만 수신하면 재전송하기 때문에 Reno 버전보다 빠른 재전송을 할 수 있다[13].

2.2 라우팅 프로토콜의 종류

기존의 유선 망과 마찬가지로 Ad-hoc 무선망에서도 어떤 라우팅 프로토콜과 트래픽을 사용하는가에 따라 망의 성능이 좌우된다. Ad-hoc 무선망에서 사용되는 라우팅 알고리즘은 유선망에서 사용되는 라우팅 알고리즘을 기반으로 개발되었다. Ad-hoc 무선망에서 사용되는 라우팅 프로토콜은 크게 Table-Driven 방식과 On-Demand 방식 두 가지 부류로 나눌 수 있다[2].

Table-Driven 프로토콜은 각각의 이동 노드가 무선망 내의 모든 경로의 정보를 유지하고 있기 때문에 경로 요구

시 최적의 경로를 설정 할 수 있다. 그러나 최신의 정보를 유지하기 위해, 제어 패킷을 통해 주기적으로 정보를 갱신해야 하므로 실제 전달하는 데이터 외에 많은 망 트래픽을 유발하는 단점을 가진다. 대표적인 Table-Driven 라우팅 프로토콜에는 DSDV(dynamic destination squence distance vector), WRP(wireless routing protocol), GSR(global state routing), FSR(fisheye state routing) 등이 있다.

On-Demand 라우팅 프로토콜은 특정 목적지에 대한 경로를 요구하였을때만 경로 설정 단계가 수행된다. 경로 설정 단계가 완료되어 새로운 경로가 발견되면 경로 유지 단계가 수행된다.

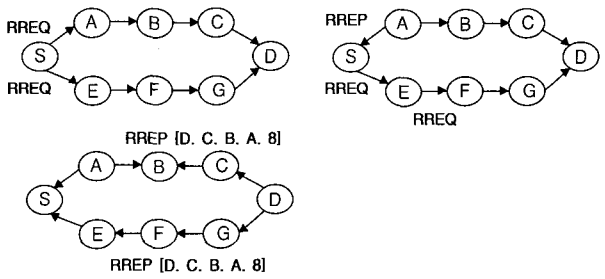
경로 유지는 목적지에게 더 이상의 요구 패킷이 없거나, 목적지까지의 경로를 사용할 수 없을 때까지 계속 수행된다. 이런 프로토콜의 단점은 경로 설정 과정에서 지연이 발생한다는 점과 홉 거리에 따른 관점에서 볼 때 최적의 경로를 보장하지 못한다는 점이다. 대표적인 On-Demand 라우팅 프로토콜에는 CBRP(cluster based routing protocol), DSR(dynamic source routing), AODV(adhoc on demand distance vectoc), TORA(temporally ordered routing algorithm) 등이 있다[3, 6].

2.3 DSR(Dynamic Source Routing) 알고리즘

DSR은 카네기 멜론 대학의 Monarch 프로젝트에 의해 제안된 On-Demand 방식의 대표적인 프로토콜로서 기본적으로 소스 라우팅의 알고리즘을 사용한다[3, 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 설명한다.

2.3.1 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 알아내어야 한다. 이 때 경로 탐색 단계를 사용한다. 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 2) DSR의 경로 설정 과정

(그림 2)는 메시지 전달 이후 소스 노드 주위의 모든 노드들은 그 패킷을 받게되고 각각의 라우터들은 자신이 가지고 있는 라우트 캐쉬의 경로를 조사한다. 그리하여 RREQ 패킷의 목적지 노드의 값이 그 노드의 라우트 캐쉬에 존재하면, 그 노드는 RREP 패킷을 반환하고 그렇지 않을 경우 수신한 RREQ 패킷을 다른 이웃 노드들에게 재전송한다. 이러한 방법을 통해 RREQ 패킷은 목적지 노드를 찾을 때까지 전체 노드에 전송하는 과정을 나타내고 있다.

이러한 작업을 거친 후에 최종적으로 목적지 노드에 패킷이 도착하면 목적지 노드는 소스 노드에게 역 방향의 경로로 RREP 패킷을 보내고, RREP 메시지에 의해 기억된 경로는 차후 데이터 전달에 사용되기 위해 소스 노드에 저장된다[4, 6].

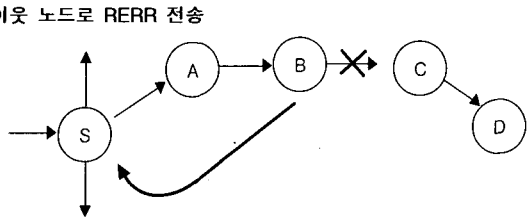
그러나 소스 노드에서 발생한 RREQ 패킷은 모든 노드에 방송되는 것이므로 같은 목적지로 두 개 이상의 경로가 들어갈 수가 있게 되고, 목적지 노드에는 소스와 목적지는 같지만 경로가 다른 RREQ 패킷들이 각기 다른 경로를 통해 여러 개 들어올 수 있다. 이 때 목적지 노드는 먼저 받은 RREQ 패킷을 저장하고 있다가 다음 RREQ 패킷에서 소스와 목적지가 같으나 경로가 다른 것이 발견되면 이후의 것을 모두 폐기한다. 왜냐하면 가장 먼저 도착한 것이 가장 빠른 경로이기 때문이다. 그렇기 때문에 소스 노드는 목적지 노드로부터 단 한 개의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

2.4.2 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 생성하고 더 이상 그 경로를 사용하지 못하도록 하기 위해 자신의 캐쉬로부터 실패한 경로 정보를 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.

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 패킷

이 전달되면 노드 A에서는 노드 S로 응답신호를 보낸다. 이와 마찬가지로 B가 패킷을 받았을 때 B는 A에게 응답신호를 보낸다.



(그림 3) 경로 유지 단계

그러나 (그림 3)에서와 같이 경로가 파손되었을 경우 노드 C는 노드 B에게 응답신호를 보내지 못하게 되고, 노드 B가 그것을 감지하면 경로 유지 알고리즘이 사용된다. 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, 경로 캐쉬에서 검색된 새로운 경로 정보로 패킷 헤더를 갱신시킨다.

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 ROUTE ERROR 메시지를 보낸다. 소스 노드 S가 노드 B로부터 ROUTE ERROR 메시지를 받으면 S는 자신의 라우트 캐쉬에 저장되어있는 노드 D로 가는 경로를 지우고 ROUTE ERROR 메시지를 이웃노드들에게 전파하여 패킷이 전달되지 않았음을 알린다[3, 4].

3. 시뮬레이션 환경

본 장에서는 모의실험에서 사용된 파라미터와 망 모델에 대하여 설명한다. 시뮬레이션 툴은 버클리 대학에서 개발한 분산 객체 네트워크 시뮬레이터인 NS-2(Network Simulator)를 사용하였다[8].

3.1 시뮬레이션 파라미터

본 논문에서 라우팅 프로토콜을 분석하기 위해 비 실시간 트래픽인 TCP 중에서 TCP Tahoe, Reno, Sack 버전을 사용하였으며, 어플리케이션으로 FTP를 사용하였다. 그리고, MAC 계층의 인터페이스로 IEEE에서 규정한 802.11을 사용하였으며, 전체 무선망의 대역폭은 2Mbps로 설정하였다.

이동 노드에서 사용하는 안테나는 Omni 안테나를 사용하였고, 이동노드의 무선 주파수 전달범위는 250m로 설정하였다.

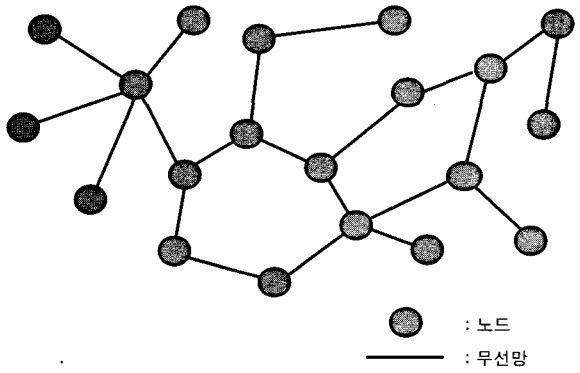
〈표 1〉 모의 실험에서 사용된 파라미터

토폴로지의 크기(m×m)	300×300	500×500	800×800
이동 속도(m/sec)	10	20	30
트래픽 소스	TCP Tahoe, Reno, Sack		
어플리케이션	FTP		
데이터 패킷 크기	512 Byte		
MAC 인터페스	802.11 (IEEE에서 규정		
안테나	Omni 안테나		
전파 전달 거리	250m		

Ad-hoc 무선망의 라우팅 프로토콜은 대표적인 On-Demand 라우팅 프로토콜중 하나인 DSR을 사용하는 Ad-hoc 무선망에서 TCP Reno버전, Sack버전, Tahoe 버전의 성능을 비교 분석하였다. 모의 실험에서 사용된 파라미터를 정리하면 <표 1>과 같다. 토폴로지 크기와 이동속도 파라미터는 <표 1> 에서와 같이 이동속도(m/sec) 10, 20, 30과 망 크기 300×300, 500×500, 800×800으로 설정한 이유는 최악의 경우를 가정으로 한 것이다.

3.2 망 모델

본 논문의 모의 실험에서는 DSR의 TCP 버전별 처리율 관계를 분석하기 위해 이동 노드의 수를 20개로 고정하여, 노드의 이동 속도를 10, 20, 30m/s로 가변 시키고 망의 크기도 300×300, 500×500, 800×800m로 가변 하였다. 그리고 TCP 데이터 패킷의 크기를 512 Bytes로 고정하였고, 100초 동안 실험하였다. 이동 노드의 유휴 시간(Pause time)은 0초로 설정하여 지속적으로 노드가 이동하는 망을 설계하였다. (그림 4)는 모의 실험에서 사용된 망의 예이다.



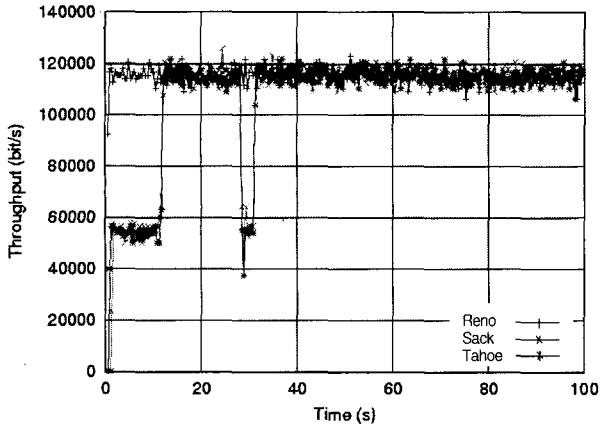
(그림 4) 모의 실험에서 사용된 망의 예

4. 시뮬레이션 결과분석

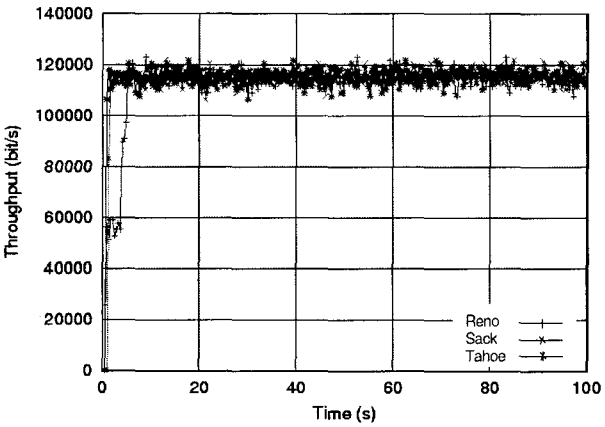
본 장에서는 TCP Tahoe, Reno, Sack를 DSR에 적용하여 수신 노드에서 처리한 패킷의 처리율을 기준으로 분석하였고, 망의 지연율은 고려하지 않았다. 그리고 처리율은 단위 시간당 목적지 노드가 수신한 패킷 양에 시간을 나눈

값을 처리율로 나타내었다.

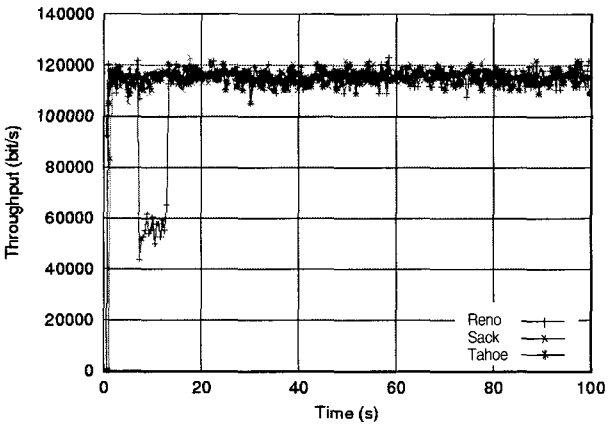
4.1 이동 노드의 속도 변화에 따른 버전별 TCP 처리율의 변화(1)



(그림 5) 노드 이동 속도가 10m/sec, 망 크기가 300×300일 때 Tahoe, Reno, Sack의 처리율



(그림 6) 노드 이동 속도가 20m/sec, 망 크기가 300×300m일 때 Tahoe, Reno, Sack의 처리율



(그림 7) 노드 이동 속도가 30m/sec, 망 크기가 300×300m일 때 Tahoe, Reno, Sack의 처리율

(그림 5), (그림 6), (그림 7)은 Ad-hoc 무선망에서 이동 노드의 이동 속도를 10, 20, 30m/sec로 증가시키고, 망의 크기를 300m×300m로 고정하였을 때 TCP Tahoe, Reno, Sack 버전에서 처리되는 패킷의 처리율을 나타낸다.

(그림 5)은 이동 노드의 수가 20개, 이동 속도를 10m/sec로 고정하였을 때, TCP Reno, Sack, Tahoe에서 처리된 패킷의 처리율을 나타낸다. 그림에서 살펴보면, Reno는 약 120 kbps의 안정적인 처리율을 보이는 반면에, Sack와 Tahoe는 모의 실험 초기, 약 15초와 28초경에 약간의 처리율 변화가 발생한다는 것을 알 수 있다. 이는 목적지 노드나 데이터를 목적지까지 중계하던 이웃 노드가 이동하면서 발생된 경로 재 설정 시간이라고 볼 수 있고, 이 처리율이 0으로 떨어지지 않고 40kbps 정도로 나타나는 것은 각각의 노드가 이미 거쳐갔던 목적지까지의 경로를 라우팅 캐쉬에 저장하였기 때문에 그만큼 빨리 경로를 재 설정하여 데이터를 전달할 수 있다는 것이다.

(그림 6)은 이동 노드의 수가 20개, 이동 속도를 20m/sec로 고정하였을 때 TCP Reno, Sack, Tahoe 버전의 처리율을 나타낸다. Reno, Sack, Tahoe 모두 경로가 설정된 이후 약 120kbps로 안정적인 처리율이 유지됨을 볼 수가 있다.

(그림 7)은 이동 노드의 수가 20개, 이동 속도를 30m/sec로 고정하였을 때, TCP 버전별 트래픽이 처리된 처리율을 나타낸다. Reno에서 8초경에 목적지 노드나 데이터를 목적지까지 중계하던 이웃 노드가 이동하면서 발생된 경로 재 설정 시간이 발생하는 것을 볼 수 있다. 그 이후 모두 경로가 설정된 뒤로 약 120kbps로 안정적인 처리율이 유지됨을 볼 수가 있다.

(그림 5), (그림 6), (그림 7)에 의해 망의 크기가 작고 이동 속도가 일정할 경우에 이동 노드의 이동 속도가 변하더라도 약간의 처리율 변화가 있기는 하지만 TCP Tahoe, Reno, Sack가 거의 안정적인 처리율을 보이고 있음을 알 수 있다.

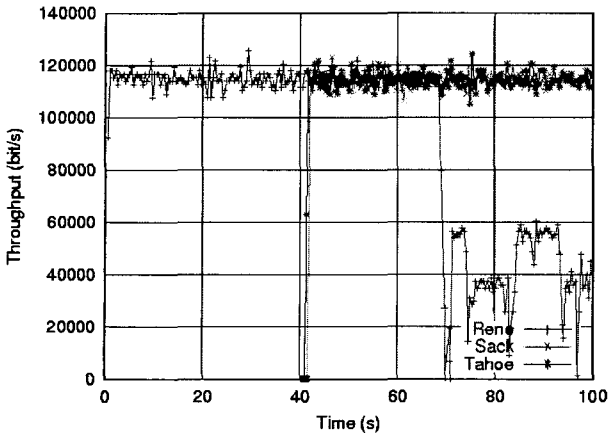
4.2 이동 노드의 속도 변화에 따른 버전별 TCP 처리율의 변화(2)

(그림 8), (그림 9), (그림 10)은 Ad-hoc 무선망에서 이동 노드의 이동 속도를 10, 20, 30m/sec로 점차 증가시키고, 망의 크기를 800m×800m로 고정하였을 때 TCP Tahoe, Reno, Sack 버전에서 처리되는 패킷의 처리율을 나타낸다.

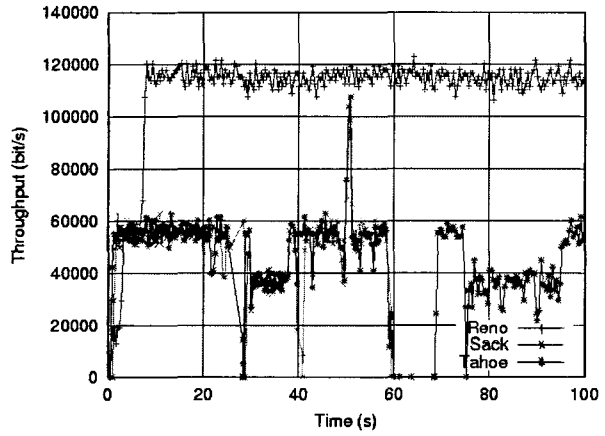
(그림 8)에서 Reno는 경로가 설정된 이후 120kbps로 안정적인 처리율이 유지됨을 볼 수가 있다. 그러나 Sack와 Tahoe에서는 데이터가 정상적으로 전소오디지 못하다가 일정 시간이 지난 후에 안정적인 처리율을 보임을 알 수 있다.

(그림 9)에서 Reno는 설정된 경로를 따라 안정적으로 패킷이 전송되다가 70초경에 패킷의 손실이 일어나 다시 연

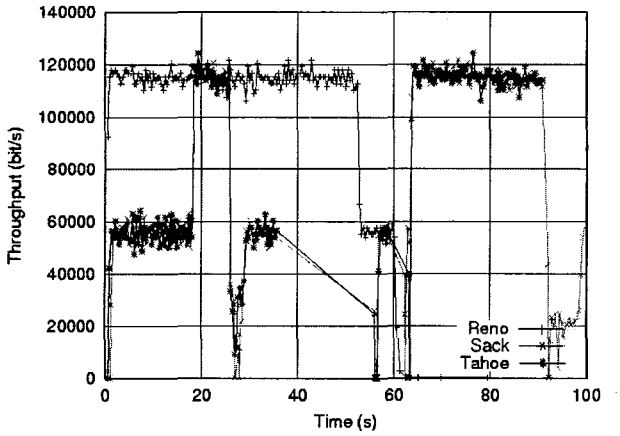
결이 재 설정되는데, Reno의 빠른 회복 알고리즘으로 패킷의 손실 발생시 이에 대한 복구 상황을 그래프로 볼 수 있다. 반면에 Sack와 Tahoe에서는 약 25초 이후로 극단적인 데이터 손실이 발생된다.



(그림 8) 노드 이동 속도가 10m/sec, 망 크기가 800×800m일 때 Tahoe, Reno, Sack의 처리율



(그림 9) 노드 이동 속도가 20m/sec, 망 크기가 800×800m일 때 Tahoe, Reno, Sack의 처리율



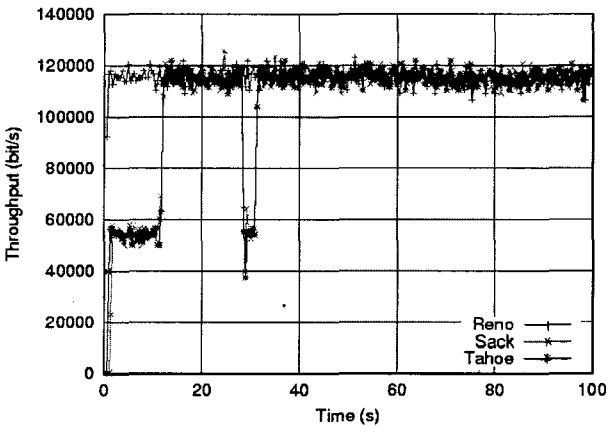
(그림 10) 노드 이동 속도가 30m/sec, 망 크기가 800×800m일 때 Tahoe, Reno, Sack의 처리율

(그림 10)에서 TCP Reno는 시뮬레이션 초기에는 안정된 처리율을 보이지만 시간이 지날수록 변화가 심하고, Tahoe와 Sack는 모의 실험 초기부터 트래픽 변화가 심하게 발생함을 알 수 있다. 이는 이동 노드의 이동 도가 빨라지면 그 만큼 패킷의 전송에 많은 변화들이 발생하여 패킷의 손실이 발생하기 때문이다. 망의 크기를 증가시키고 이동 노드의 이동 속도를 변화시키는 시뮬레이션 결과 Reno가 Sack나 Tahoe보다 비교적 처리율이 안정적으로 나타나는 모습을 볼 수 있었고 이동 속도도 빨라지고 망도 커지게 되면 Sack와 Tahoe에서 처리율의 변화가 많이 나타나는 것을 볼 수 있다.

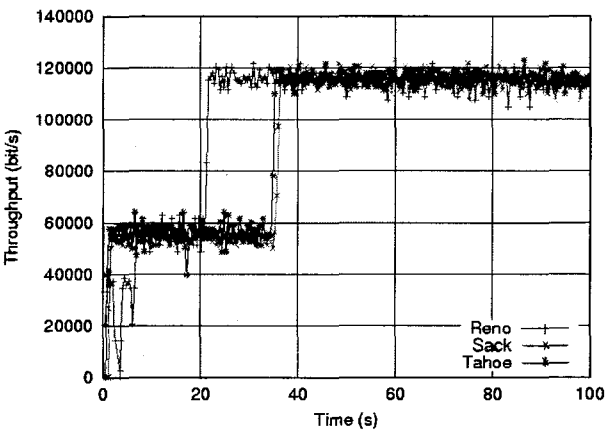
4.3 망의 크기 변화에 따른 TCP의 처리율 변화

(그림 11), (그림 12), (그림 13)은 Ad-hoc 무선망에서 이동 노드의 이동 속도를 10m/sec로 고정하고, 망의 크기를 300m ×300m, 500m×500m, 800m×800m로 가변 시켰을 때 처리율을 나타낸 것이다.

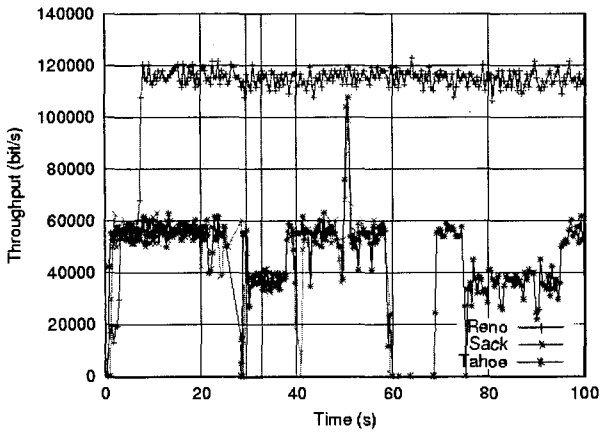
망의 크기가 300m×300m일 때에 거의 안정적인 처리율을 나타났으나, Tahoe와 Sack는 모의 실험 초기에 약간의 처



(그림 11) 노드 이동 속도가 10m/sec, 망 크기가 300×300m일 때 Tahoe, Reno, Sack의 처리율



(그림 12) 노드 이동 속도가 10m/sec, 망 크기가 500×500m일 때 Tahoe, Reno, Sack의 처리율



(그림 13) 노드 이동 속도가 10m/sec, 망 크기가 800×800m일 때 Tahoe, Reno, Sack의 처리율

리율의 변화가 발생함을 알 수 있다. 망의 크기가 500m×500m로 커진 경우, 경로설정 시간이 Reno는 약 20초, Sack와 Tahoe는 약 37초 경과하여 처리율이 안정적으로 변화한다는 것을 알 수 있다.

망의 크기가 800×800으로 변화되었을 때, Reno는 안정적인 트래픽을 보이는 반면, Tahoe와 Sack는 전체 모의 실험 시간에 걸쳐 처리율이 0으로 떨어지는 극단적인 변화가 발생한다는 것을 알 수 있다.

이동 노드의 이동 속도를 10m/s로 고정시키고, 망의 크기를 변화시켰을 때 Reno에서는 모두 120kbps로 안정적인 처리율을 보이는 반면, Sack와 Tahoe는 처리율의 변화가 심하고, 60kbps 정도로 처리율이 훨씬 낮다는 것을 알 수 있다.

5. 결 론

최근 몇 년 동안 Ad-hoc망에서 다양한 라우팅 프로토콜이 연구되어 왔다. 그러나 이러한 프로토콜들을 이용하였을 때, 데이터를 전달하는 프로토콜에 대한 성능에 관한 연구가 미진했다.

본 논문에서는 TCP Reno, Sack, Tahoe 버전과 분산객체 네트워크 시뮬레이터인 NS-2를 사용하여 Ad-hoc Routing Protocol들 중에서 On-Demand 방식인 DSR을 이용하여 TCP 버전별 트래픽 성능을 비교 분석하였다.

본 논문에서 시뮬레이션은 노드의 이동 속도와 망의 크기를 변화 시켜 실험하였다. 성능 평가요소로는 시간에 따른 처리율을 사용하였으며, 지연은 고려하지 않았다.

모의 실험 결과 Reno 버전의 경우 망의 크기나 노드의 이동 속도가 변하여도 거의 안정적인 처리율을 보였다. 그러나 Tahoe와 Sack 버전에서는 망의 크기나 노드의 이동 속도가 낮을 때 안정된 처리율을 보였지만, 망의 크기나 노드의 이동 속도가 증가하면 할수록 Reno 버전보다 처리율과 그 변화가 심하다는 것을 알 수 있었다. 즉, TCP Reno 버전은 노드의 이동속도와 망의 크기에 그다지 큰 영향을 받지 않지만, TCP Tahoe와 Sack는 이동 속도와 망 크기에 따라 민감한 반응을 보였다.

결론적으로 DSR 라우팅 프로토콜을 사용하는 Ad-hoc 무선망에서, TCP를 사용하는 경우, Reno 버전을 사용하는 것이 안정적인 데이터 전송이 된다는 것을 알 수 있었다.

참 고 문 헌

[1] S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET) : Routing Protocol Performance Issues and Evaluation Considerations," rfc 2501, January, 1999.

[2] H Singh, S Singh, "Energy consumption of TCP Reno, Newreno, and SACK in multi-hop wireless networks," ACM SIGMETRICS Performance Evaluation, pp.206-216, June, 2002.

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

[4] Elizabeth M. Royer, Chai-Keoeng Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks," IEEE, April, 1999.

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

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

[7] 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.

[8] IEEE standard Department, "Wireless Lan Medium access control(MAC) and physical layer(PHY) specifications," IEEE standard 802.11, 1997.

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

[10] IEEE standard Department, "Wireless Lan Medium Access Control(MAC) and physical layer(PHY) specifications," IEEE standard 802.11, 1997.

[11] H. Wang, H. Xin, D. S. Reeves, and K. Shin, "A Simple Refinement of Slow-Start of TCP Congestion Control," IEEE Symposium on Computers and Communications, Antibes France, pp.1-8, 2000.

[12] Kyu-Nam Lee, "The study of TCP Performance over DSR," thesis for master degree, Aug., 2002.

[13] K. Fall and S. Floyd, "Comparisons of Tahoe, Reno, and Sack TCP," December, 1995.

박 승 섭

e-mail : parkss@dolphin.pknu.ac.kr

1982년 경북대학교 공과대학 전자계산전공 (공학사)

1984년 일본대학 이공학연구과(공학석사)

1984년~1986년 한국통신 연구원

1989년~1990년 일본동북대학 객원교수

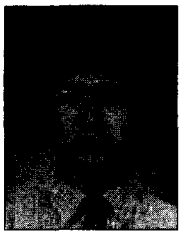
1993년 일본 동북(Tohoku)대학(공학박사)

1998년 Philippine Ateneo de davao university, visiting prof.

1998년~1999년 부경대학교 컴퓨터멀티미디어 공학부장

1986년~현재 국립 부경대학교 전자컴퓨터정보통신공학과 교수

관심분야 : 인터넷 엔지니어링, 초고속통신망, 멀티미디어



육 동 철

e-mail : net607@mail1.pknu.ac.kr

1998년 동서대학교 컴퓨터공학과 졸업 (공학사)

2000년 부경대학교 전자계산학과 졸업 (이학석사)

2002년 부경대학교 전자계산학과 박사 수료

관심분야 : 컴퓨터 네트워크, XML 전자상거래, OR-Mapper 관리시스템