이동 애드혹 네트워크에서의 TCP-Vegas 성능향상 기법

준회원 배 한 석*, 송 점 기**, 정회원 김 동 균***, 박 정 수****, 김 형 준*****

Improving TCP-Vegas Performance over Mobile Ad-hoc Networks

Hanseok Bae*, Jeomki Song** *Associate Members*, Dongkyun Kim***, Jung-Soo Park****, Hyoung-Jun Kim***** *Regular Members*

요 약

애드혹 네트워크에서의 종단 응용프로그램간의 신뢰성 있는 데이터 전송과 인터넷과의 자연스러운 접속을 가능하기 위해서는 기존의 유선 인터넷에서 사용되고 있는 TCP의 수용이 바람직하다. 따라서 최근의 연구에서 애드혹 네트워크에 가장 적합한 TCP의 변이종을 찾기 위해 다양한 경로설정 프로토콜 상에서 TCP의 성능평가가 이루어져왔다. 하지만 OLSR상에서의 TCP의 성능평가는 이루어지지 않고 있다. 따라서 본 논문에서는 IETF에서 표준화가 된 AODV와 OLSR상에서 현재 유선망에 가장 널리 사용되고 있는 TCP-Reno와 과거의 연구에서 TCP-Reno보다 우수한 성능을 보인 TCP-Vegas의 성능을 NS-2 시뮬레이터를 사용하여 비교 분석하였다. 실험결과 어떠한 경로설정 프로토콜을 선택하느냐에 따라 TCP의 성능이 큰 차이를 보였다. 그리고 측정된 RTT를 기반으로 하여 전송률을 조절하는 TCP-Vegas를 경로가 빈번하게 변경되는 애드혹 네트워크에 적용할 경우 부정확한 Base RTT로 인해 TCP-Vegas의 성능감소를 초래한다. 따라서 본 논문에서는 이러한 문제를 해결하기 위해 경로가 변경되었을 시 새로이 Base RTT를 측정하는 방법을 제안하였다.

Key Words : MANET, proactive, reactive, TCP-Vegas, route change

ABSTRACT

TCP is needed as a transport protocol to provide reliable end-to-end message delivery for MANETs in order to achieve a smooth integration with the fixed Internet. Particularly, TCP has its variants, namely TCP-Reno and TCP-Vegas. However, there has been no research work on extensive performance comparison of TCP-Reno and TCP-Vegas over AODV and OLSR. This paper is the first trial to perform the research by using ns-2 simulator. Through the extensive simulations, we found that which to select among routing protocols is more important than which to select among TCP variants, because the performance difference between TCP-Reno and TCP-Vegas over any selected routing protocol is not so much outstanding. Particularly, TCP-Vegas relies on an accurate BaseRTT estimation in order to decide the sending rate of a TCP Sender. However, it cannot be directly applied to MANET because a route change makes the Base RTT used over a previous path obsolete. Therefore, we propose a technique for improving the performance of TCP-Vegas by considering the route change, and show the performance improvement through simulation study.

* 경북대학교 컴퓨터공학과 (bae@monet.knu.ac.kr), ** 경북대학교 컴퓨터공학과 (jksong@monet.knu.ac.kr),
*** 경북대학교 컴퓨터공학과 (dongkyun@knu.ac.kr), **** 한국전자통신연구원 (pjs@knu.ac.kr)
***** 한국전자통신연구원(khj@etri.re.kr)
논문번호 : KICS2005-09-381, 접수일자 : 2005년 9월 21일, 최종논문접수일자 : 2006년 1월 21일

Ⅰ. 서 론

이동 애드혹 네트워크는 고정된 인프라가 없는 환경에서 제한된 전송 범위를 가지는 이동 노드들로 구성된 네트워크를 말한다. 이러한 이동 노드들은 데이터를 송수신하는 호스트로서의 역할뿐만 아니라 다른 노드들의 데이터를 중계하는 라우팅 기능도 수행한다.

무선 통신에 기반을 둔 애드혹 네트워크는 유선 네트워크와 비교할 때 다음과 같은 특징을 갖는다. 첫째, 노드의 이동으로 인하여 네트워크 토폴로지가 지속적으로 변화한다. 노드는 일반적으로 사람이 휴대할 수 있을 정도의 크기이며 망 내를 자유롭게 이동한다. 둘째, 노드들은 배터리와 무선 전송 기술을 사용하기 때문에 자원사용에 있어서 제약이 크다. 즉, 유선망의 노드에 비해 상대적으로 적은 양의 메모리, 낮은 연산 능력, 한정된 배터리 전력을 가진다. 마지막으로 적은 대역폭과 높은 비트 에러율을 갖는 무선 링크를 사용한다.

위와 같은 특성으로 인하여 기존 유선 네트워크에서의 경로설정 프로토콜을 애드혹 네트워크에 그대로 적용하는 데에는 많은 문제점이 따르게 된다. 따라서 1990년대 후반에는 IETF(Internet Engineering Task Force) 산하에 MANET(Mobile Ad-hoc NETworks) 워킹그룹[1]을 형성하여 이동 노드들 간의 높은 통신 효율성을 제공하기 위한 경로설정 프로토콜에 대한 연구를 활발히 진행하였으며 그 결과, AODV(Ad Hoc On-Demand Distance Vector)[2]와 OLSR(Optimized Link State Routing)[3]이 경로설정 프로토콜로 표준화 되었다.

OLSR과 DSDV(Destination-Sequenced Distance Vector)[4]와 같은 Proactive 경로설정 프로토콜은 네트워크상의 모든 노드들이 주기적인 경로정보의 교환을 통해 네트워크상의 모든 노드로의 경로를 사전에 유지한다. 이에 반해 AODV와 DSR(Dynamic Source Routing)[5]과 같은 Reactive 경로설정 프로토콜은 특정 노드로의 데이터 전송요구가 발생하면, 경로탐색 과정을 통해 목적노드로의 경로를 획득하는 방법이다.

앞서 언급한 네트워크 계층의 프로토콜 이외에도 종단 응용프로그램 간의 신뢰성 있는 데이터 전송을 보장하기 위해 애드혹 네트워크에 적합한 전송(Transport) 계층에 관한 연구가 필요하다. 하지만 인터넷과의 자연스런 접속을 가능하게 하기 위해서는 기존의 유선 인터넷에서 사용되고 있는 TCP (Transmission Control Protocol)의 수용이 바람직하다. 따라서 최근의 연구에서 애드혹 네트워크에 가장 적합한 TCP의 변이를 찾기 위해 다양한 경로설정 프로토콜 상에서 TCP의 성능 평가가 이루어져 왔다. 하지만 대부분의 연구들은 하나의 경로설정 프로토콜 상에서 여러 가지 TCP 변이들의 성능을 평가하였거나 다양한 경로설정 프로토콜 상에서 선택된 하나의 TCP의 성능평가가 이루어져왔다. 최근에 들어서는 Reactive 경로설정 프로토콜인 AODV와 Proactive 경로설정 프로토콜인 DSDV상에서 여러 가지 TCP 변이들의 성능평가가 이루어졌다[6]. 하지만 IETF에서 표준화가 된 OLSR상에서의 TCP의 성능평가는 아직까지 이루어지지 않고 있다. 따라서 본 논문에서는 AODV와 OLSR상에서 현재 유선망에서 가장 널리 사용되고 있는 TCP-Reno[7]와 과거 유선망에서의 연구에서 TCP-Reno보다 우수한 성능을 보인 TCP-Vegas[8]의 성능을 비교 분석하였다. 또한 측정된 RTT를 기반으로 전송률을 조절하는 TCP-Vegas는 경로 재설정으로 인해 발생하는 Base RTT의 부정확성이 성능감소의 요인이 되는데, 이러한 Base RTT의 부정확성 문제를 경로가 빈번하게 변경되는 OLSR상에서 목적노드로의 경로가 변경되었을 시 새로이 측정하는 방법을 제안하였다.

본 논문의 2장에서는 본 논문에서 사용한 경로설정 프로토콜과 TCP 변이들을 소개한다. 3장에서는 AODV와 OLSR상에서 TCP-Reno와 TCP-Vegas의 성능을 비교 분석하였으며, 4장에서는 본 논문에서 제안하는 TCP-Vegas의 성능 향상 알고리즘에 대해 기술하고 시뮬레이션을 통해 성능을 평가한다. 마지막으로 5장에서는 결론을 맺는다.

Ⅱ. 관련 연구

애드혹 네트워크를 위한 다양한 경로설정 프로토콜이 제안되었으나 본 장에서는 본 논문에서 사용한 AODV와 OLSR에 대해서만 언급한다. 그리고 현재 유선네트워크에서 가장 널리 사용되는 TCP-Reno와 과거 연구에서 TCP-Reno보다 우수한 성능을 보인 TCP-Vegas에 대해서 논한다.

2.1 AODV

AODV는 대표적인 Reactive 경로설정 프로토콜로서 소스노드가 데이터 전송이 필요할 때, 즉 트래픽이 발생하는 시점에서 목적노드까지의 경로를 찾

는다. 소스노드가 목적노드로까지의 경로가 필요한 경우 경로를 탐색하기 위해 RREQ(Route REQuest) 메시지를 전송하며, 이때 각 RREQ는 Broadcast ID를 포함한다. 이는 중복된 메시지의 재전송을 회피하는데 그 목적이 있다. RREQ를 수신한 노드는 먼저 소스노드의 주소와 Broadcast ID의 검사를 통해 패킷의 처리여부를 결정한다. 만약 이미 수신한 패킷이라면 별도의 처리과정 없이 폐기하고 수신하지 않은 패킷이라면 송신노드로의 역방향 경로 정보를 경로설정 테이블에 저장한다. 그리고 자신이 목적노드가 아니라면 RREQ를 이웃노드로 재전송 한다. 이 때 만약 RREQ를 수신한 노드가 목적노드로의 경로정보를 가진 중간노드, 또는 RREQ의 목적노드일 경우 소스노드로 RREP(Route REPly)메시지를 응답한다. 이때 RREP는 RREQ 처리과정에 생성된 역경로를 통해 유니캐스트로 송신노드까지 전송되게 되며, RREP를 수신한 각 노드들은 목적노드로의 경로를 설정한다. 이러한 경로 설정과정을 통해 소스노드는 목적노드로의 데이터를 전송 할 수 있게 된다.

소스노드가 목적노드로 데이터를 전송하는 동안 노드의 이동으로 인해 경로가 단절되었을 경우에는, 이를 탐지한 노드가 지역복구(Local Recovery)를 수행하거나 소스노드로 RERR(Route ERRor) 메시지를 전달함으로써 소스 노드가 경로 재설정을 수행하도록 한다[2].

2.2 OLSR

OLSR은 Proactive 경로설정 프로토콜로서, 유선 네트워크에서 사용되는 Link State 프로토콜을 애드혹 네트워크 환경에 맞게 최적화한 프로토콜이다. 대부분의 Proactive 경로설정 프로토콜은 경로설정 테이블을 유지하기 위해 주기적으로 링크정보 메시지를 네트워크의 모든 노드에게 blind 플러딩을 한다. 하지만 OLSR은 MPR(Multi-Point Relay)이라 불리는 특정 노드만이 링크정보 메시지를 재전송함으로서 불필요한 중복된 메시지의 수를 줄인다. 네트워크의 모든 노드들은 이웃노드들과 주기적으로 Hello 메시지를 교환하고, 이 메시지를 통해 각 노드들은 MPR Set을 구성한다. 이때 Hello 메시지에는 해당노드가 도달 가능한 이웃노드들의 정보가 포함된다. MPR을 구성하는 절차는 다음과 같다.

1) 1홉 거리에 있는 노드들 중 2홉 노드에 도달하기 위해 유일한 경로를 제공하는 노드들을 MPRs에 추가한다.
2) 2홉 거리에 있는 노드들 중 현재의 MPRs을 통해 도달 가능한 노드들을 2홉 노드 set에서 제외한다.
3) MPRs에 포함되지 않은 1홉 노드들 중 2홉 거리에 있는 노드들을 가장 많이 포함하는 노드를 MPRs에 추가한다.
4) 2홉 거리에 있는 노드들이 모두 포함될 때까지 2)의 과정부터 반복한다.

경로설정 테이블을 구성하기 위해서 각 노드들은 유선 네트워크에서의 Link State 경로설정 프로토콜과 같이 주기적으로 링크정보를 모든 노드들에게 전송한다. 하지만 OLSR은 Link State 프로토콜처럼 자신의 모든 링크 정보를 메시지에 포함하는 것이 아니라 자신을 MPR로 선택한 노드들(MPR Selector set)만을 포함하여 링크정보 메시지의 크기를 줄인다. 또한 모든 노드가 링크정보를 전송하는 것이 아니라 특정노드의 MPR로 선택된 노드만이 전송함으로서 네트워크에 플러딩 되는 링크정보 메시지의 수를 줄인다. 각 노드들은 네트워크상의 노드들로부터 수신한 네트워크의 토폴로지 정보에서 Dijkstra 알고리즘을 이용하여 모든 노드로의 경로를 경로설정 테이블에 유지한다. 이때 경로의 양단을 제외한 나머지는 MPRs로만 구성된다[3].

2.3 TCP-Reno

TCP-Reno는 현재 유선네트워크에서 가장 널리 사용되는 TCP로서 혼잡제어는 Slow Start, Congestion Avoidance, Fast Recovery로 구성되어 있다.

Slow Start는 데이터 전송이 시작 되었을 때와 시간초과(Time-out)가 발생했을 때 실행된다. Slow Start 구간에서는 CWND 크기를 지수적으로 증가시키다가 임계값에 이르면 Congestion Avoidance 상태에 돌입하며, 이때 임계값은 바로 이전의 혼잡상황이 발생했을 때의 CWND 크기의 반을 임계값으로 사용한다. 그리고 Congestion Avoidance 상태에서의 CWND의 크기는 Slow Start 구간과 다르게 선형으로 증가시킨다.

TCP-Reno에서 손실된 패킷의 재전송은 시간초과와 중복된 ACK에 의해서 수행된다. 전송한 패킷에 대한 시간초과가 발생하면 즉, 해당 패킷에 대한 ACK을 RTO시간 내에 수신하지 못할 경우 이를 심각한 통신망 혼잡 현상이 발생하였다고 판단하고 CWND를 하나의 세그먼트 크기로 설정하고 Slow

Start 구간으로 돌입한다. 이에 반해 빠른 재전송(Fast retransmit)은 순서에 맞지 않은 TCP 세그먼트가 수신 될 때 수신측은 비어 있는 TCP 세그먼트에 대한 재전송 요청을 계속 송신노드에게 하는데, 이때 동일한(Duplicated) ACK 메시지 3개를 송신노드에서 수신하게 되면 전송된 TCP 세그먼트가 손실 되었다고 생각하고 RTO이 발생하기 전에 재전송을 수행한다. 그리고 통신망의 혼잡 가능성을 생각하여 전송률을 반으로 줄인다[7].

2.4 TCP-Vegas

TCP-Vegas는 패킷 손실이 생길 때까지 CWND를 증가시키는 TCP-Reno와는 달리 네트워크에 혼잡이 발생하기 전에 미리 CWND를 감소시켜 패킷 손실을 방지하는 혼잡 회피 방법(Congestion Avoidance mechanism)을 사용한다. TCP-Vegas의 CWND 크기는 측정된 RTT 값을 기반으로 계산된 Expected와 Actual의 차이 Diff에 의해 결정된다. 연결 상태에서의 모든 RTT 측정값 중 최소값을 Base RTT로 지정하고 이 값을 기준으로 기대할 수 있는 전송률 Expected을 계산한다. 그리고 현재의 실제 전송률 Actual은 현재 RTT 측정값을 기준으로 계산 한다 (식 1).

$$Expected = WindowSize/BaseRTT \quad (1)$$
$$Actual = WindowSize/RTT$$
$$Diff = (Expected - Actual) \times BaseRTT$$

Vegas는 α와 β 두개의 임계값을 정의한다. 만약에 Diff가 α보다 작다면 네트워크 상황이 좋다고 판단하고 현재의 CWND에서 1을 증가시키고 β보다 크다면 네트워크에 혼잡이 발생할 것이라고 판단하고 CWND 1을 감소시킨다. 이 외에 Diff값이 α와 β의 사이에 있을 경우는 현재의 CWND를 유지한다. 또한 Vegas는 손실된 패킷의 재전송을 위해 각 패킷에 대한 fine grained RTO를 유지한다. 이는 중복된 ACK을 수신 했을 때 해당 패킷에 대한 시간초과 발생여부의 기준이 되며, 만약 시간초과가 발생했다면 해당 패킷이 손실 되었다고 판단하고 추가적인 중복된 ACK 없이 빠른 재전송을 한다[8].

Ⅲ. 애드혹 경로설정 프로토콜 상에서의 TCP-Reno와 TCP-Vegas 성능분석

본 장에서는 TCP-Reno와 과거의 유선 네트워크 상에서의 연구에서 TCP-Reno보다 우수한 성능을 보인 TCP-Vegas를 Reactive 경로설정 프로토콜의 대표적인 프로토콜인 AODV와 Proactive 경로설정 프로토콜의 대표적인 프로토콜인 OLSR상에서 성능을 시뮬레이션을 통해 비교 분석하였다. 성능 분석을 위해 NS-2 시뮬레이터를 사용하였다[9]. 본 논문에서 사용된 NS-2 시뮬레이터는 무선 환경과 노드의 이동성을 지원하는 CMU Wireless Extension이 포함되어 있다[10].

3.1 시뮬레이션 환경

애드혹 네트워크를 구성하기 위해 1000m x 1000m 크기의 평면 내에 50개의 노드를 랜덤하게 배치하였으며, 노드들의 이동성을 위해 Random waypoint 모델을 사용하였다. 이 모델에 의하면 각 노드는 임의의 목적지를 선택하고 0에서 최대속도 값 사이의 임의의 속도로 목적지를 향해 이동한다. 노드가 해당 목적지에 도달하면 일정시간 동안 머물러 있다가 또 다른 임의의 목적지를 선택해서 이동한다.

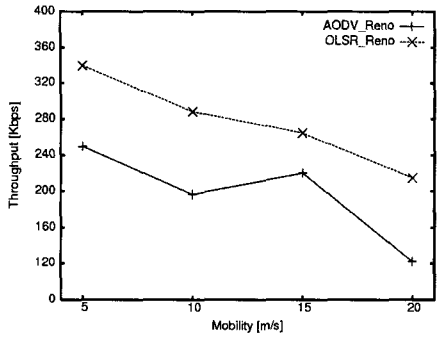
MAC 계층으로는 802.11을 사용하였으며, 각 노드들의 전송범위는 250m이며 대역폭은 2Mbps이다. 경로설정 프로토콜로는 AODV와 OLSR을 사용했으며, 전송 프로토콜로는 TCP-Reno와 TCP-Vegas를 사용하였다. 그리고 처리량 측정을 위해 FTP 트래픽 생성기를 사용하였다.

시뮬레이션 결과의 신뢰성을 확보하기 위해 20개의 서로 다른 시나리오를 사용하여 평균값을 측정하였으며, 각 시나리오의 실행시간은 500초이다. TCP 연결은 임의의 두 노드에 하나의 연결을 설정하였으며, 백그라운드 트래픽을 위해 5개의 UDP 연결을 사용하였다.

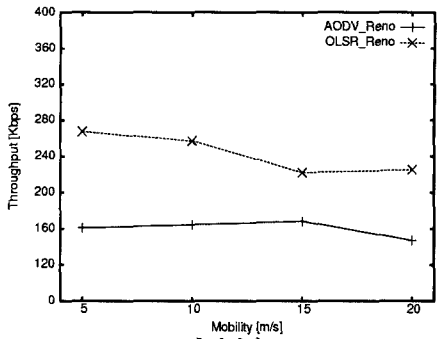
3.2 성능평가 및 분석

첫 번째로, AODV와 OLSR상에서 TCP-Reno의 처리량을 노드의 이동성에 따라 측정하였다. 측정결과 모든 이동속도에서 OLSR 상에서의 TCP-Reno가 AODV 상에서의 TCP-Reno보다 높은 처리량을 보였다 (그림 1).

OLSR이 AODV보다 TCP-Reno의 처리량이 많은 이유로는 경로 단절에 대한 복구과정의 차이를 들 수 있다. AODV는 목적노드까지의 경로가 단절되면 새로운 경로를 찾기 위해서 경로 재설정 과정을 실행한다. 이때 경로 재설정 과정의 지연으로 인해 TCP는 특정 세그먼트에 대한 시간초과를 겪게 되고, 이를 네트워크의 혼잡으로 오인하고 CWND



(a) 휴지시간 0



(b) 휴지시간 20

그림 1. AODV상에서의 TCP-Reno와 OLSR상에서의 TCP-Reno의 처리량 비교

의 크기를 극단적으로 떨어뜨린다. 이에 반해 OLSR의 각 노드들은 자신의 링크 정보가 바뀌게 되면 즉각적으로 바뀐 정보를 전체 네트워크에 플러딩 하고, 이를 수신한 각 노드들은 경로설정 테이블을 새롭게 작성한다. 이 결과 OLSR에서의 각 노드들은 다른 모든 노드들에 대한 가장 최신의 경로 정보를 유지하게 되어 특정 노드로 가는 경로가 단절되었을 시 빠른 복구가 가능하다. 이러한 이점으로 인해 OLSR상에서의 TCP는 AODV상에서의 TCP보다 경로단절에 의한 시간초과 이벤트가 적게 발생하게 되며, 그 결과 불필요한 CWND 크기의 감소횟수를 줄일 수 있다.

그리고 경로복구 중에 소스노드는 목적노드로의 새로운 경로를 찾을 때까지 더 이상의 TCP 세그먼트를 전송 할 수 없게 되는데, 이는 경로복구 시간이 길어질수록 TCP의 처리량이 더욱더 감소하게 됨을 의미한다. 그림 2에서 보듯이, AODV는 경로재설정 시간의 지연으로 인해 TCP 세그먼트를 전송할 수 없는 반면, OLSR은 빠른 복구로 인해 TCP 세그먼트를 계속적으로 전송하는 것을 알 수 있다. OLSR은 특정 노드로까지의 여러 가지 경로 중 가장 적은 홉 수를 가지는 경로를 경로설정 테이블에 유지한다. 이에 반해, AODV는 목적지에 가장 먼저 도착한 RREQ에 대해서 RREP를 응답하는데, 이는 소스노드로부터 목적노드로까지의 가장 적은 홉 수를 가지는 경로를 선택하는 것을 보장하지 못한다. TCP의 처리량은 하위계층인 무선 링크의 특성상 소스노드와 목적노드의 홉 수가 길어질수록 급격히 감소하는데, 목적노드로까지의 가장 적은 홉 수를 유지하는 OLSR이 그렇지 않은 AODV보다 좋은 성능을 보인다. 아래 그림 3은 전송되는 TCP 세그먼트의 순서번호(바이트단위)를 트레이스 한 결과이다. OLSR상에서의 TCP-Reno가 AODV상에서

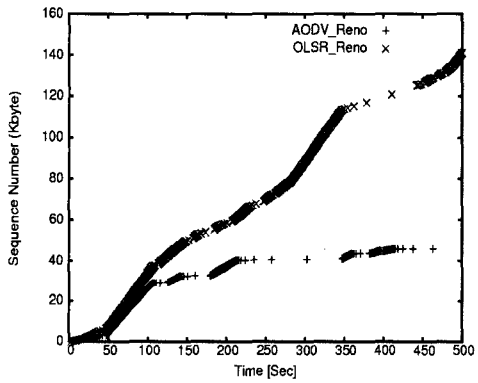


그림 2. AODV상에서의 TCP-Reno와 OLSR상에서의 TCP-Reno의 CWND 크기 비교

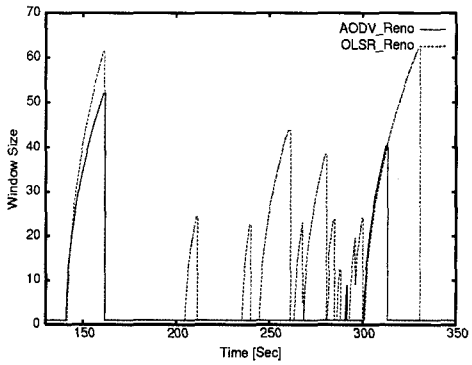
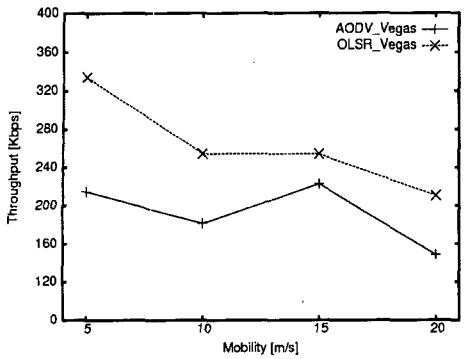


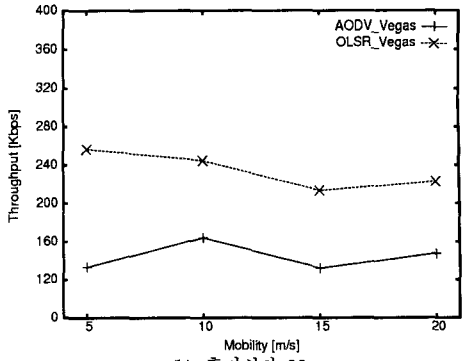
그림 3. AODV상에서의 TCP-Reno와 OLSR상에서의 TCP-Reno의 순서번호 트레이스

의 TCP-Reno보다 빠르게 진행하는 것을 알 수 있으며, AODV상에서의 TCP-Reno는 잦은 경로단절로 인해 시간초과가 OLSR상에서의 TCP-Reno보다 많이 발생함을 알 수 있다.

TCP-Vegas를 OLSR과 AODV상에서 실험한 결과, 앞서 설명한 경로설정 프로토콜의 특성으로 인해 위의 TCP-Reno의 결과와 유사한 결과를 보였다

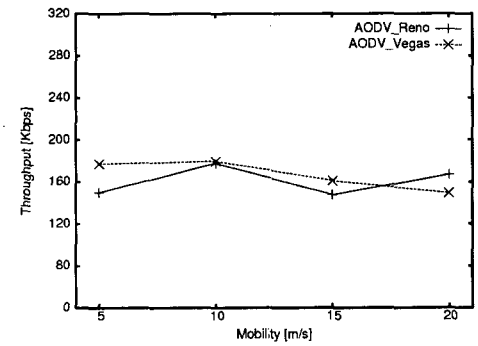


(a) 휴지시간 0

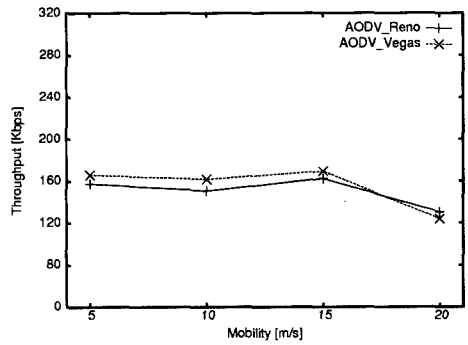


(b) 휴지시간 20

그림 4. AODV상에서의 TCP-Vegas와 OLSR상에서의 TCP-Vegas의 처리율 비교



(a) 휴지시간 0



(b) 휴지시간 20

그림 5. AODV상에서의 TCP-Reno와 TCP-Vegas의 처리율 비교

(그림 4).

두 번째로, 하나의 경로설정 프로토콜 상에서 TCP-Reno와 TCP-Vegas의 처리량을 노드의 이동성에 따라 측정하였다. 측정결과 Reactive 경로설정 프로토콜인 AODV상에서의 처리량은 TCP-Vegas가 TCP-Reno보다 나은 처리량을 보였다(그림 5).

TCP-Reno는 적극적으로 CWND를 증가시키는 반면, TCP-Vegas는 비교적 작은 CWND를 유지하기 때문에 숨은 노드 문제(hidden node problem)에 의해 발생할 수 있는 불필요한 경로 재설정 과정을 줄일 수 있다[11]. 링크계층에서는 프레임 전송을 위해 시도횟수(Retry Limit) 만큼 시도하고도 해당 프레임을 전송하지 못하면, 링크가 단절되었다고 판단하고 상위계층으로 경로단절을 통보한다. 불필요한 CWND의 증가는 공유된 무선자원의 혼잡을 발생시키고, 이러한 혼잡은 데이터 전송의 계속적인 실패를 야기하는데, 이를 링크계층에서는 경로가 유지되어 있음에도 불구하고 경로 단절로 오인하고 상위계층으로 통보하게 된다. 그 결과 경로설정 프로토콜은 불필요한 경로 재설정 과정을 수행하게 된다.

따라서 적절한 크기의 CWND를 유지함으로 불필요한 경로 재설정 과정을 줄이게 되는 TCP-Vegas가 TCP-Reno 보다 나은 결과를 보였다. 그림 6에서 보듯이, 비록 TCP-Reno가 초반에는 빠르게 진행하지만 많은 데이터 유입으로 인해 불필요한 경로 재설정과정을 발생시키게 되고, 그 결과 전체적인 TCP의 성능은 떨어짐을 알 수 있다.

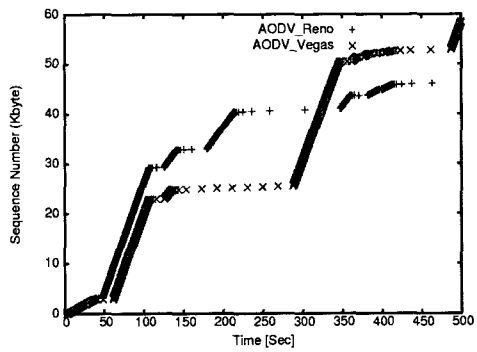
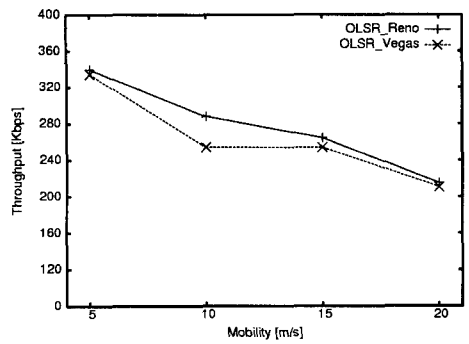
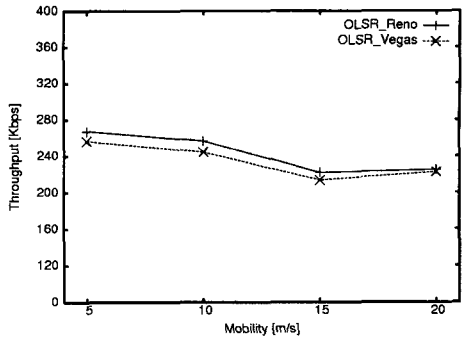


그림 6. AODV상에서의 TCP-Reno와 TCP-Vegas의 순서번호 트레이스

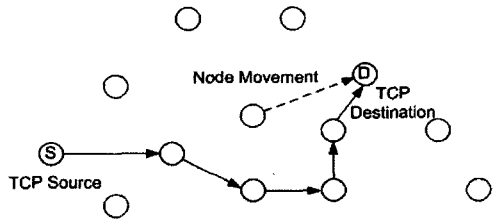


(a) 휴지시간 0

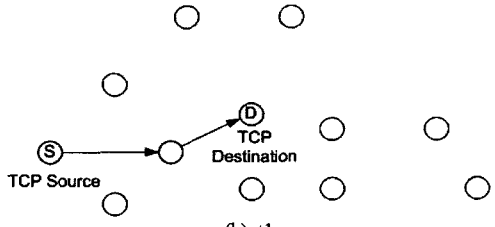


(b) 휴지시간 20

그림 7. OLSR상에서의 TCP-Reno와 TCP-Vegas의 처리율 비교



(a) t0



(b) t1

그림 8. 경로변경의 예

이에 반해 Proactive 경로설정 프로토콜인 OLSR 상에서는 TCP-Reno가 TCP-Vegas 보다 뛰어난 성능을 보였다 (그림 7). 윈도우 기반의 TCP-Reno와는 달리 TCP-Vegas의 전송률은 측정된 RTT값에 따라 조절되어진다. TCP-Vegas는 데이터 전송 중의 가장 작은 RTT를 Base RTT로 지정하고 이 값을 기준으로 Expected를 계산한다. 이 Expected를 Actual과 비교하여 전송률을 조절하는데, 노드의 이동성으로 인해 경로 재설정이 빈번한 애드혹 네트워크에서는 현재 전송 중인 경로의 Base RTT가 아니라 이전 경로들 중 제일 작은 RTT를 Base RTT로 유지한다. 이러한 Base RTT의 부정확성은 TCP-Vegas의 성능을 저하시키는 요인이 된다. AODV는 목적노드로까지의 경로를 찾으면 경로상의 특정 노드의 이동으로 인해 그 경로가 단절 될 때까지 같은 경로를 유지하는 반면, OLSR에서의 각 노드들은 토폴로지가 변할 때마다 경로설정 테이블을 새로이 계산한다. 이 결과 OLSR은 AODV보다 빈번하게 경로 재설정이 일어나게 되고, 이는 Base RTT의 정확성을 더욱 떨어트리는 요인이 된다. 그러므로 RTT측정값에 따라 전송률을 조절하는 TCP-Vegas는 목적노드까지의 경로가 빈번하게 바뀌는 OLSR보다 경로가 단절될 때까지 같은 경로를 유지하는 AODV에서 높은 성능을 보였다.

Ⅳ. 애드혹 네트워크에서의 TCP-Vegas 성능향상을 위한 알고리즘

측정된 RTT를 기반으로 전송률을 조절하는 TCP-Vegas는 연결시간 동안 경로의 변경이 일어나지 않는 것을 가정하고 있다. 하지만 TCP-Vegas를 애드혹 네트워크에 적용할 경우에는 노드들의 이동으로 인해 경로 재설정이 되었음에도 불구하고 TCP 송신측은 이 사실을 알 수 없기 때문에 예전경로의 Base RTT로 전송률을 조절하는 문제점을 가지고 있다.

경로 재설정으로 인한 Base RTT의 부정확성 문제는 TCP-Vegas의 성능을 떨어트리는 요인이 되며, 토폴로지가 자주 변경되는 환경에서는 더욱 심각해진다. 그림 8과 같이 t0 시간에 소스노드 S와 목적노드 D간의 경로가 t1 시간에 노드의 이동으로 인해 경로가 변경되었을 경우 두 노드간의 홉 수가 늘어났음에도 불구하고 TCP-Vegas는 이전의 홉 수가 적은 경로의 Base RTT를 유지하는 문제점을 가지고 있다.

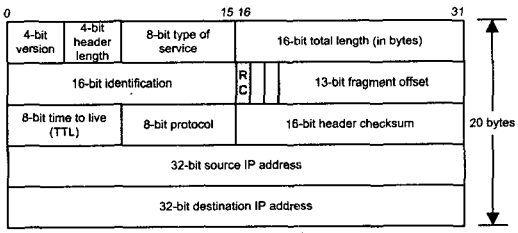
앞의 실험 결과, AODV상에서의 TCP-Vegas는 TCP-Reno보다 나은 성능을 보인 반면, AODV보다 경로가 자주 변경되는 OLSR상에서는 TCP-Vegas가 TCP-Reno보다 좋지 못한 성능을 보였다. 따라서

본 논문에서는 Base RTT의 부정확성으로 인해 발생하는 TCP-Vegas의 성능감소를 해결하기 위해 소스노드에서 목적노드로의 경로가 빈번하게 변경되는 OLSR상에서 경로가 변경되었을 시 Base RTT를 새롭게 설정하는 방법을 제안한다.

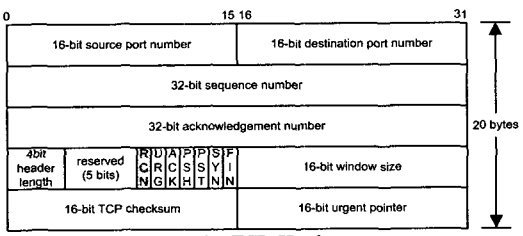
4.1 TCP-Vegas의 성능향상을 위한 알고리즘

Base RTT의 재계산을 위해 TCP 연결상의 중간 노드는 목적노드까지의 경로가 변경되었음을 탐지하면, 소스노드로 경로가 변경되었다는 정보를 특별한 메시지를 통해 전달하고, 이를 수신한 소스노드는 Base RTT를 현재 경로상의 RTT로 재설정 하면 된다. 하지만 이는 소스노드로 경로가 변경되었다는 정보를 알리기 위해 추가적인 메시지를 요구한다. 따라서 본 논문에서는 추가적인 메시지 없이 전송중인 세그먼트의 IP 헤더와 TCP 헤더의 필드에 경로가 변경되었다는 정보를 추가함으로서 Base RTT의 부정확성 문제를 해결하는 방법을 제안한다.

전송경로상의 각 노드들은 경로설정 테이블에 Route_Changed 필드를 유지하는데, 이는 목적노드로의 경로변경 탐지에 사용되어진다. 전송경로상의 각 노드들은 목적노드로의 다음 홉이 변경되었을 때 Route_Changed 필드를 1로 설정한다. 각 노드들은 목적노드로의 패킷을 전송할 때 다음 홉에 대한 Route_Changed 필드가 1로 설정되어 있으면 경로가 바뀌었다고 판단하고 IP 헤더에 RC(Route Change) 플래그를 1로 설정한다(그림 9(a)). 그리고 다음 패킷에 대한 플래그 설정을 막기 위해 경로설정 테이블의 Route_Changed 필드를 0으로 설정한다.



(a) IP Header



(b) TCP Header

그림 9. 추가 필드

Algorithm 1 Intermediate Nodes

```
if ( Route_Changed is set in the Routing Table ) then
    if ( RC Flag is not set in the IP Header) then
        Set the RC flag;
    end if
end if
transmit the packet;
```

그림 10. 중간 경로상의 노드들이 수행하는 알고리즘

Algorithm 2 TCP Receiver Node

```
if ( Packet Received ) then
    if ( RC Flag is set in the IP Header) then
        Set the RC flag in the ACK packet;
    end if
    transmit the packet;
end if
```

그림 11. TCP 수신노드가 수행하는 알고리즘

Algorithm 3 TCP Sender Node

```
if ( ACK Received ) then
  if ( RC Flag is set ) then
      Base_RTT = Current_RTT;
  else
      Normal Processing;
  end if
end if
```

그림 12. TCP 송신노드가 수행하는 알고리즘

RC 플래그가 설정된 세그먼트를 수신한 목적노드는 이를 소스노드로 알리기 위해 해당 세그먼트에 해당하는 ACK에 RCN(Route Change Notification) 플래그를 설정해서 소스노드로 전송한다. 이때 RCN 플래그는 TCP 헤더에 예약된 플래그를 이용한다(그림 9(b)). 소스노드는 RCN이 설정된 ACK을 수신하면 목적노드로의 경로가 변경되었음을 인식하고 현재의 RTT값을 Base RTT로 설정한다. 그림 10, 11, 12는 본 논문에서 제안한 방법을 의사코드로 나타낸 것이다. 에 위치하고 있다면 반대로 위쪽에 0mm 아래쪽에 5mm의 여백을 주어야 한다.

4.2 실험 결과 및 평가

첫 번째로, 이동성이 높은 환경에서 본 논문에서 제안하는 방법과 기존의 TCP-Vegas를 비교하기 위해 휴지시간이 0일 때 노드의 이동성에 따른 처리량을 측정하였다. 그림 13에서 볼 수 있듯이 이동성이 높을수록 TCP의 처리량은 감소함을 알 수 있다. 또한 노드의 이동성이 비교적 낮을 때는 본 논문에서 제안하는 방법과 기존의 TCP-Vegas의 성능차이는 크게 나지 않지만, 노드의 이동성이 높을수록 두 방법의 성능차이가 많이 남을 알 수 있다. 이는 노드의 이동성이 높을수록 소스노드와 목적노드간의 경로 변경이 빈번하게 일어나게 되는데 본

논문에서 제안하는 방법은 경로가 변경될 때마다 새로이 Base RTT를 설정하는 반면 기존의 TCP-Vegas는 예전 경로들의 Base RTT중 가장 작은 값을 계속적으로 유지하기 때문이다. 그림 14는 전송되는 TCP 세그먼트의 순서번호(바이트단위)를 트레이스 한 결과이다. 본 논문에서 제안하는 방법이 기존의 TCP-Vegas 보다 빠르게 진행 하는 것을 알 수 있다.

두 번째로 비교적 노드의 이동성이 적은 환경에서의 본 논문에서 제안하는 방법과 기존의 TCP-Vegas를 비교하기 위해 휴지시간이 20일 때 노드의 이동성에 따른 처리량을 측정하였다. 노드의 이동성이 높을 때 보다는 성능차이가 많이 나지 않지만 모든 이동속도에서 본 논문에서 제안하는 방법이 우수함을 알 수 있다(그림 15). 그리고 휴지시간이 0일 때와 마찬가지로 노드의 이동성이 높을수록 성능차이가 많이 남을 알 수 있다. 또한 순서번호의 트레이스 결과 본 논문에서 제안하는 방법이 빠르게 진행함을 알 수 있다(그림 16).

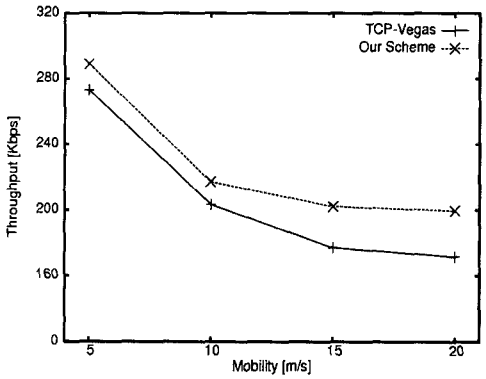


그림 13. 이동성이 높은 환경에서의 처리량 비교

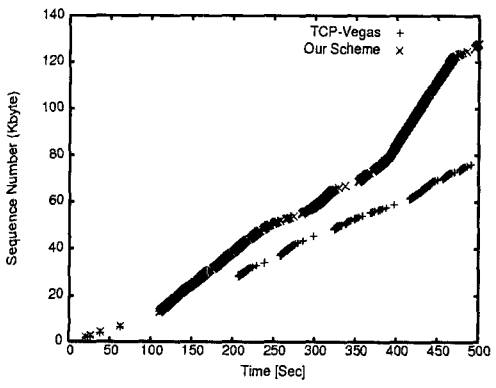


그림 14. 이동성이 높은 환경에서의 순서번호 트레이스

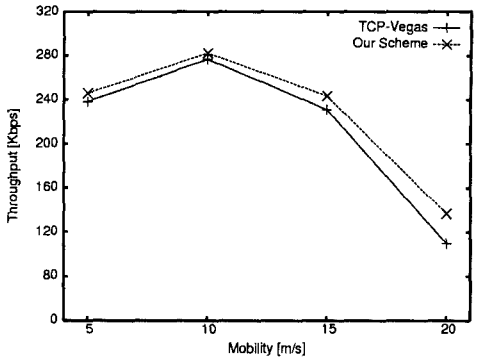


그림 15. 이동성이 낮은 환경에서의 처리량 비교

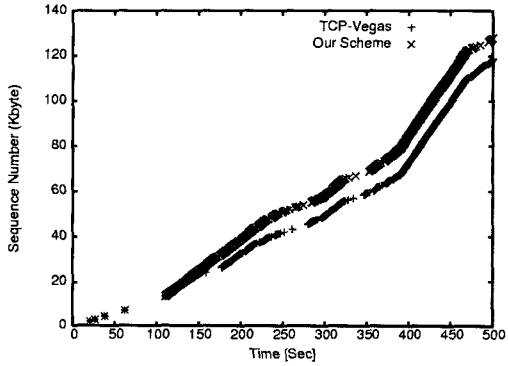


그림 16. 이동성이 낮은 환경에서의 순서번호 트레이스

V. 결 론

측정된 RTT를 기반으로 하여 전송률을 조절하는 TCP-Vegas를 경로가 빈번하게 변경되는 애드혹 네트워크에 적용할 경우 부정확한 Base RTT로 인해 TCP-Vegas의 성능향상을 기대하기 힘들다. 따라서 본 논문에서는 이러한 문제를 해결하기 위해 경로가 변경되었을 시 새로이 Base RTT를 측정하는 방법을 제안하였다. 실험결과, 노드의 이동성이 높을수록 본 논문에서 제안한 방법이 기존의 TCP-Vegas보다 우수한 성능을 보였다.

비록 본 논문에서는 경로가 빈번하게 변경되는 Proactive 경로설정 프로토콜에 적용하였지만, Reactive 경로설정 프로토콜에도 적용할 수 있다. 대부분의 Reactive 경로설정 프로토콜은 단절된 경로의 복구를 위해 소스노드가 직접 경로를 재탐색하거나 중간경로상의 노드가 지역복구를 수행한다. 소스노드가 직접 경로를 재탐색할 경우 TCP-Vegas 송신노드는 하위 경로설정 프로토콜로부터의 통보에 의해 Base_RTT를 새로이 설정 할 수 있다. 하지만 중간경로상의 노드가 지역복구를 수행 할 경우에는 TCP-

Vegas 송신노드는 경로가 변경되었다는 정보를 지역복구를 수행하는 노드로부터의 통보에 의해서 가능하다. 하지만 이는 추가적인통보메시지를 요구한다. 따라서 지역복구를 수행하는 노드는 본 논문에서 제안한 방법을 적용하여 목적노드로의 새로운 경로를 획득한 후, 처음으로 전송하는 패킷에 route_changed 플래그를 설정하여 전송함으로써 TCP-Vegas 송신노드로의 경로변경 정보를 통보할 수 있다. 본 논문에서 제안한 방법을 Reactive 경로설정 프로토콜에 적용 하였을 경우에도 보다 나은 성능을 기대 할 수 있다.

그리고 최근 IETF MANET 워킹 그룹에서 표준화 한 AODV와 OLSR 경로설정 프로토콜 상에서 TCP-Reno와 TCP-Vegas의 성능을 비교 분석하였다. 시뮬레이션 결과를 종합하면, TCP 처리량 측면에서 OLSR이 TCP의 종류에 상관없이 AODV보다 나은 성능을 보였다. 그러나 TCP-Vegas는 OLSR상에서보다 AODV상에서의 성능이 우수했다. 그리고 경로설정 프로토콜이 TCP의 성능에 큰 영향을 미침을 알 수 있다.

참 고 문 헌

[1] Internet Engineering Task Force, "Manet working group charter," http://www.ietf.org/html.charters/manet-charter.html

[2] C. E. Perkins, E. M. Belding-Royer, and S. R. Das, "Ad-hoc On-Demand Distance Vector (AODV) Routing", IETF RFC 3561.

[3] T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized Link State Routing Protocol (OLSR)", IETF RFC 3626.

[4] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing(DSDV) for Mobile Computers", SIGCOMM Symposium on Communications Architectures and Protocols, September 1994.

[5] D. B. Johnson, D. A. Maltz, and Y. Hu, "The Dynamic Source Routing Protocol for Mobile Ad-hoc Networks (DSR)", IETF Internet Draft, draft-ietf-manet-dsr-08.txt, February 2003.

[6] S. Papanastasiou, M. Ould-Khaoua, "Exploring the performance of TCP Vegas in Mobile Ad hoc Networks", International Journal of Communications Systems, pp. 163-177, Vol. 17, Issue 2, March 2004.

[7] V. Jacobson, "Congestion avoidance and control," SIGCOMM Symposium on Communications Architectures and Protocols, Aug 1988

[8] L. Brakmo and L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," IEEE Journal on Selected Areas in Communications, vol. 13, pp. 1465-1480, October 1995.

[9] K. Fall and K. Varadhan, "NS notes and documentation," the VINT Project, UC Berkely, LBL USC/ISI, and Xerox PARC, Available from http://www.isi.edu/nsnam/ns/, November 1997.

[10] CMU Monarch Group, "CMU Monarch extension to the NS-2 simulator," Available from http://monarch.cscmu.edu/cmu-ns.html, 1998.

[11] Fu Z, Zerfos P, Luo H, Lu S, Zhang L and Gerla M, "The impact of multihop wireless channel on TCP throughput and loss," IEEE INFOCOM, 2003.

배 한 석 (Hanseok Bae) 준회원

2003년 경일대학교 컴퓨터공학과(학사)

2005년 경북대학교 컴퓨터공학과(공학석사)

<관심분야> Mobile Ad Hoc Network, TCP over MANET, SCTP

송 점 기 (Jeomki Song) 준회원

2004년 경일대학교 컴퓨터공학과(학사)

2006년 경북대학교 컴퓨터공학과(공학석사)

<관심분야> Mobile Ad Hoc Network, 자동 네트 워킹 기술, IPv6

김 동 균 (Dongkyun Kim) 정회원

1994년 경북대학교 컴퓨터공학과(학사)
1996년 서울대학교 컴퓨터공학과(공학석사)
2001년 서울대학교 전기·컴퓨터공학부(공학박사)
1999년 미국 Georgia Institute of Technology, 방문 연구원
2002년 미국 University of California at Santa Cruz, Post-Doc. 연구원
2003년 3월~현재 경북대학교 컴퓨터공학과 조교수
<관심분야> 이동인터넷, 초고속 인터넷, Mobile Ad Hoc Network, 무선 LAN 등

박 정 수 (Jung-Soo Park) 정회원

1992년 경북대학교 전자공학과(학사)
1994년 경북대학교 전자공학과(공학석사)
1994년 2월~현재 한국전자통신연구원 선임연구원
2005년 3월~현재 경북대학교 전자공학과 박사과정
<관심분야> IPv6, Ad-hoc, Security, WSN

김 형 준 (Hyoung-Jun Kim) 정회원

1986년 광운대학교 컴퓨터공학과(학사)
1988년 광운대학교 컴퓨터공학과 석사
1988년 2월~현재 한국전자통신연구원 팀장
2000년 3월~현재 충남대학교 컴퓨터과학과 박사과정
<관심분야> IPv6, RFID/USN