이동성이 큰 에드-혹 네트워크를 위한 효율적인 멀티캐스트 라우팅 프로토콜

(An Efficient Multicast Routing Protocol for Ad-Hoc Networks with High Mobility)

김 예 경 † 이 미 정 ††
(Yekyung Kim) (Meejeong Lee)

요 약 본 논문에서는 에드-혹 망의 멀티캐스트 라우팅 프로토콜인 ODMRP(On-Demand Multicast Routing Protocol)를 확장한 PatchODMRP를 제안한다. ODMRP는 네트워크 상에서 멀티캐스트그룹의 송신원으로부터 수신원에 이르는 경로 상에 있는 노드들을 FG(Forwarding Group) 노드로 선출하여 이들이 해당 멀티캐스트그룹에 속하는 패킷을 모두 플러딩하도록 함으로써 멀티캐스트 그룹 데이타 전송을 담당하는 메쉬를 구성하도록 하는 방안이다. 그런데 ODMRP는 주기적으로 이 메쉬를 구성하는 FG 노드들을 재 선정하기 때문에 이 주기가 길어지면 메쉬 구성이 네트워크 노드들의 이동성을 따라가지 못해 메쉬 분리가 발생하고 데이타가 손실될 수 있다. 반면에 이 주기를 짧게 하면 오버헤드가 지나치게 커질 수 있다. 특히, 송신원의 수가 적은 경우 ODMRP의 메쉬는 매우 성기게 형성되는데, 이 때 호스트들의 이동성이 크면 메쉬 연결을 유지하기 위하여 이 주기를 짧게 잡아주거나, 높은 데이타 손실율을 감수해야 한다. 본 논문에서는 이 문제점을 해결하고자 각 FG 노드들이 BEACON 신호를 이용해 자신에 인접한 메쉬에 손실이 발생한 것을 인지하고 이를 국부적인 플러딩을 통하여 빠르게 복구하는 메커니즘을 ODMRP에 추가한 PatchODMRP 방식을 제안한다. 시뮬레이션을 통하여 기존의 ODMRP와 제안하는 PatchODMRP의 성능을 비교한 결과, PatchODMRP가 호스트의 이동성에 훨씬 강하며, ODMRP에 비하여 낮은 오버헤드로 높은 데이타 전송률을 제공할 수 있음을 보여주었다.

Abstract In this paper, we propose an ad-hoc multicast routing protocol, referred to as Patch ODMRP. PatchODMRP extends the ODMRP (On-Demand Multicast Routing Protocol), which is a *mesh-based multicast routing protocol proposed for ad-hoc networks.* In ODMRP, the nodes that are on the shortest paths between the multicast group members are selected as Forwarding Group (FG) nodes, and form a forwarding mesh for the multicast group. The ODMRP reconfigures the forwarding mesh periodically to adapt it to the node movements. When the number of sources in the multicast group is small, usually the forwarding mesh is formed sparsely and it can be very vulnerable to mobility. In this case, very frequent mesh reconfigurations are required in ODMRP, resulting in large control overhead. To deal with this problem in a more efficient way, PatchODMRP makes each FG node keep checking if there is a symptom of mesh separation around itself. When an FG node finds such symptom, it tries to patch itself to the mesh with a local flooding of control messages. Through a course of simulation experiments, the performance of PatchODMRP is compared to the performance of ODMRP. The simulation results show that PatchODMRP improves the data delivery ratio, and reduces the control overheads. It has also been shown that the performance gain is larger when the degree of node mobility is bigger

· 본 연구는 학술진흥재단의 2000년도 BK21 사업 지원으로 수행되었음
† 학생회원 : 이화여자대학교 컴퓨터학과
982COG30@mm.ewha.ac.kr
†† 정 회 원 : 이화여자대학교 컴퓨터학과 교수
lmj@mm.ewha.ac.kr
논문접수 : 2000년 8월 8일
심사완료 : 2000년 12월 26일

1. 서 론

에드-혹 망은 고정된 네트워크 기반이 없이 이동하는 호스트들로 구성되는 다중 홉 무선 네트워크이다. 이러한 에드-혹 망의 대표적인 응용으로는 재난 시 이를 해

결하기 위한 긴급 대책위원회 멤버들간의 통신이나 전투시 전투요원들간의 통신, 컨퍼런스나 야외 오락에서 멤버들 간의 통신등을 들 수 있다. 이러한 응용들은 대부분 다중점 대 다중점의 통신을 요구하기 때문에 이를 효율적으로 지원하기 위해서는 멀티캐스트 라우팅 기술이 필요하다. 그런데, 에드-혹 망은 망의 기반 구조가 예측할 수 없이 동적으로 변해 나가며, 호스트의 프로세싱 능력이 매우 제한적이라는 특성이 있어 이러한 환경에 효율적인 멀티캐스트 라우팅 프로토콜을 설계하는 것은 매우 도전적이다.

최근 에드-혹 망의 효율적인 멀티캐스팅을 위해 많은 새로운 프로토콜들이 다양하게 연구되고 있다. 이러한 프로토콜들을 크게 분류하자면, 송신원으로부터 각 수신원에 대해 유일한 최단 경로가 결정되어 이를 통해 데이타를 전달하는 트리 기반(Tree-based) 방식과 하나 이상의 경로를 통하여 데이타를 전달하는 메쉬 기반(Mesh-based) 방식이 있다. 트리 기반의 프로토콜로 제안된 것으로는 AMRoute(Adhoc Multicast Routing)[1]와 AMRIS(Ad hoc Multicast Routing protocol utilizing Increasing id-numberS)[2]등이 있고, 메쉬 기반의 프로토콜로 제안된 것 중 대표적인 것으로는 ODMRP (On-Demand Multicast Routing Protocol)[3],[4],[5]와 CAMP (Core-Assisted Mesh Protocol)[6] 등이 있다. ODMRP는 멀티캐스트 그룹의 송신원마다 메쉬를 만드는 반면, CAMP는 멀티캐스트 그룹의 공유 메쉬를 만들어 데이타를 전송한다. 이들 프로토콜들의 성능을 비교한 최근 논문에 의하면, 전반적으로 메쉬 기반의 프로토콜이 트리 기반의 프로토콜에 비해 데이타 전송률이 높으며, 제어 패킷과 중복되는 데이타 패킷의 오버헤드도 낮은 것으로 나타났다[7]. 또한, 메쉬 기반 프로토콜들 중에서는 ODMRP가 CAMP보다 호스트의 이동 정도가 커지더라도 높은 데이타 전송률을 유지하고 프로토콜의 효율면에서도 더 우수함을 볼 수 있었다[7].

그런데 ODMRP의 경우 각 송신원으로부터 모든 수신원에 이르는 최단 경로 상에 있는 노드들의 합집합으로 메쉬를 구성하기 때문에 멀티캐스트 그룹의 송신원 수가 적은 경우에는 데이타 전송을 담당하는 FG 노드들이 상대적으로 적게 선출되고, 데이타 전달 메쉬가 성글게 형성된다. 이러한 경우 FG 노드들 중 일부가 메쉬를 구성하는 이웃 FG 노드들의 전송 범위 밖으로 이동해 버리면 그 부분의 데이타 전송 경로가 끊어져 수신원까지의 데이타 전송이 이뤄지지 못하게 되기가 쉽다. ODMRP에서는 데이타 전달을 담당하는 FG 노드들의 이동으로 인한 데이타 손실을 막기 위해 노드들의 이동성이 증가하면 JOIN QUERY 패킷의 발생 빈도를 높여 자주 메쉬를 재구성할 것을 권장하고 있다. 그러나 JOIN QUERY 패킷은 망 전체에 플러딩되기 때문에 JOIN QUERY 패킷의 발생 빈도를 높이면 프로토콜 오버헤드가 급격히 증가하게 된다.

ODMRP의 이러한 단점을 극복하고자, 본 논문에서는 국부적인 경로 손실이 발생한 경우 FG 노드가 이를 감지하고 국부적으로 경로를 재 설정하는 방안인 Patch ODMRP를 제안한다. PatchODMRP는 기존 ODMRP에 손실된 경로의 국부적인 재 설정 메카니즘을 추가한 것으로, 플러딩을 요구하는 JOIN QUERY의 발생 빈도를 높이지 않고도 빠르게 경로 손실을 임시로 복구하는 것이 가능하도록 해준다. 시뮬레이션을 통하여 기존 ODMRP와 비교한 결과 송신원의 수가 적고 노드의 이동성이 높은 경우, PatchODMRP에 의하여 데이타의 전송률이 높아지고, 프로토콜의 오버헤드가 낮아지는 결과를 볼 수 있었다.

본 논문은 다음과 같은 순서로 구성된다. 먼저, 1장의 서론에 이어, 2장에서는 ODMRP에 대한 개략적 설명과 문제점을 제시한다. 그리고, 3장에서는 제안하는 PatchODMRP의 상세한 동작 방법을 설명하고, 4장에서는 기존 ODMRP와 PatchODMRP의 성능을 비교하기 위한 시뮬레이션 및 그 결과를 제시한다. 마지막으로 5장에서는 결론을 맺는다.

2. ODMRP 동작 방법

본 장에서는 제안하는 Patch ODMRP 방식을 설명하기 위해 필요한 기본적인 ODMRP의 작동 방식을 설명한다. ODMRP는 에드-혹 망에서의 멀티캐스팅을 위해 고안된 온디맨드 라우팅 프로토콜로써, 멀티캐스트 그룹의 멤버들을 연결하는 최단 경로상에 있는 노드들을 FG 노드로 선출하고 이들이 해당 멀티캐스트 그룹에 속하는 데이타를 플러딩하도록 하는 메쉬 기반 방식이다.

표 1 JOIN QUERY 패킷 형식

Type(01)	Reserved	Time to live	홉 카운트
멀티캐스트 그룹 IP 주소			
일련 번호			
송신원 IP주소			
이전 홉 IP 주소			
이전 홉 X 좌표			
이전 홉 Y 좌표			
이전 홉 이동 속도		이전 홉 이동 방향	
최소 연결 만기 시간			

표 2 JOIN REPLY 패킷 형식

Type(02)	Count	R	F	Reserved
멀티캐스트 그룹 IP 주소				
이전 홉 IP 주소				
일련 번호				
송신원 IP 주소[1]				
NEXT_HOP_IP 주소[1]				
경로 만기 시간[1]				
. . . . .				
송신원 IP 주소[n]				
NEXT_HOP_IP 주소[n]				
경로 만기 시간[n]				

데이타 전달 경로인 메쉬를 구성하는 FG 노드들을 유지하는 제어 메시지는 두 가지로, 송신원이 주기적으로 네트워크 전체에 플러딩하는 *JOIN QUERY*와 *JOIN* QUERY에 대한 응답 메시지로 수신원이 송신원에게 전달하는 JOIN REPLY가 있다. 다음의 [표 1]과 [표 2]는 각각 JOIN QUERY와 JOIN REPLY 의 패킷 형식이다.

ODMRP는 일반적인 온디맨드 유니캐스트 라우팅 프로토콜과 유사하게 요구와 응답 두 단계로 이루어진다. 먼저, **요구 단계**에서는 송신원이 자신의 정보를 담은 *JOIN QUERY* 패킷을 전체 네트워크에 플러딩한다. 네트워크의 각 호스트는 JOIN QUERY를 받으면, 먼저 메시지 캐쉬를 이용하여 패킷 중복을 체크한다. 이 중복 체크는 패킷의 루핑을 막기 위한 것으로, 호스트가 어떤 패킷을 받든지 항상 행한다. 받은 JOIN QUERY가 중복이 아니라면, [표 3]과 같은 구조를 가지는 호스트의 라우팅 테이블에 JOIN QUERY의 송신원 ID와 이 패킷을 전달한 전 홉 ID를 저장한다. 이렇게 함으로써 각 호스트는 *JOIN QUERY* 플러딩 과정에서 송신원으로의 경로를 파악하게 되고 이 정보는 추후 수신원으로부터 송신원으로 향하는 JOIN REPLY를 전달하는데 이용된다. 이렇게 JOIN QUERY 송신원으로의 라우팅 정보를 저장하고 난 후 호스트는 JOIN QUERY를 자신의 이웃 호스트에게 다시 플러딩한다. 송신원은 자신이 멀티캐스트 데이타를 전송하는 한, 정기적으로 JOIN QUERY 패킷을 발생시킴으로써 노드 이동에 맞추어 메쉬를 재구성해 나간다.

표 3 라우팅 테이블 구조

목적지 (JOIN QUERY의 송신원)	다음 홉 (JOIN QUERY를 전달해준 전 홉)

표 4 포워딩 테이블 구조

멀티캐스트 그룹 ID	타이머

한편, 수신원이 JOIN QUERY 패킷을 받으면 **응답 단계**가 시작된다. 수신원은 자신이 참여하는 멀티캐스트 그룹의 송신원으로부터 JOIN QUERY 패킷을 받으면 그에 대한 응답으로 JOIN REPLY 패킷을 만들어 이웃 호스트들에게 브로드캐스트 한다. 이 *JOIN REPLY* 패킷의 NEXT_HOP_ID 필드에는 송신원으로 가는 다음 홉 ID를 기록한다. JOIN REPLY를 받은 호스트는, 받은 JOIN REPLY 패킷의 NEXT_HOP_ID가 자신의 ID와 같은지 체크한다. 만일 틀리다면, 받은 패킷을 버리고, 같다면 [표 4]와 같은 구조를 가지는 포워딩 테이블에 해당 멀티캐스트 그룹 ID에 대한 FG_FLAG를 TRUE로 설정한다. 그리고 포워딩 테이블의 타이머를 지정하고, 타이머가 만기될 때까지 해당 멀티캐스트 그룹의 FG 노드로 작동하게 된다. 이렇게 새로이 선출된 FG 노드는 자신의 라우팅 테이블에서 JOIN REPLY의 목적지인 해당 멀티캐스트 그룹의 송신원으로 가는 다음 홉을 찾아서, 받은 JOIN REPLY 패킷의 NEXT_HOP_ID를 이 값으로 수정하고 이를 이웃 호스트들에게 브로드캐스트 한다. 이렇게 NEXT_HOP_ID가 수정되면서 JOIN REPLY가 송신원까지 전달되는 과정을 통해 *JOIN* REPLY를 발생시킨 수신원과 *JOIN* REPLY의 목적지인 송신원 사이에 해당 멀티캐스트 그룹의 데이타 전송을 담당하는 FG 노드들이 선출된다.

메쉬가 구성된 상태에서 멀티캐스트 데이타 전송은 간단하다. 송신원이 이웃 호스트들에게 멀티캐스트 데이타를 브로드캐스트하면, 이를 받은 이웃 호스트중 FG 노드들만이 이를 다시 이웃 호스트에게 브로드캐스트한다. 이러한 과정을 반복하면서, FG 노드 메쉬를 통해 데이타가 수신원까지 전달된다. 작동 방법이 플러딩과 유사하지만, FG노드들에 의해서만 플러딩되는 제한적인 플러딩이어서 순수한 플러딩에 비하여 오버헤드가 적다.

한편, ODMRP는 멀티캐스트 그룹의 참여/탈퇴를 위해서는 이를 위한 어떤 특정 패킷을 발생하지 않고, 암묵적인 참여/탈퇴 방식인 소프트 상태(soft-state) 방식을 사용한다. 송신원의 경우, 참여시 주기적으로 JOIN QUERY 패킷을 계속 보내며, 탈퇴시에는 더 이상 JOIN QUERY 패킷을 보내지 않는다. 수신원의 경우, 참여시 JOIN QUERY에 대한 응답으로 JOIN REPLY를 송신원에게 보내지만 탈퇴시에는 JOIN QUERY에

대해 더 이상 JOIN REPLY를 보내지 않는다. 수신원은 멀티캐스트 그룹에 해당하는 모든 송신원에 대한 정보를 멤버 테이블에 보관하여, JOIN QUERY를 받을 때마다 이를 갱신한다. 만일 일정 시간이 경과한 후에도 JOIN QUERY를 받지 못하면 멤버 테이블에서 해당 엔트리를 삭제한다.

3. PatchODMRP

본 장에서는 ODMRP 메쉬에 경로 손실이 발생한 경우, 이를 감지하고 국부적으로 이를 수정해 주도록 ODMRP를 확장한 PatchODMRP를 소개한다. ODMRP에서는 FG 노드들이 메쉬를 형성하므로 일반적으로 하나 이상의 경로가 존재하고 한 경로가 손실되어도 다른 경로로 데이타가 전달된다고 가정한다. 그러나, 멀티캐스트 송신원 수가 적어 멀티캐스트 그룹을 위해 선출된 FG 노드 수가 적은 경우, 호스트의 이동 속도가 크다면 메쉬의 연결성을 보장하기 위해서는 잦은 메쉬 업데이트, 즉 빈번한 JOIN QUERY 플러딩이 있어야 한다. 그러나, 이렇게 JOIN QUERY 플러딩이 늘어나면 제어 패킷 오버헤드가 증가하게 될 뿐 아니라 이로 인한 채널 경쟁으로 데이타 손실이 유발 될 수 있다.

이를 해결하고자 제안하는 PatchODMRP는 MAC 계층의 BEACON 메시지를 사용하여 송신원으로의 다음 홉인 FG 노드가 없어진 것을 감지하고, 이러한 일이 발생한 경우에는 메쉬에서의 연결에 손실이 발생했을 수 있다고 가정하고 국부적으로 이를 해결한다. 경로 갱신은 빠르게 진행되며, 메쉬가 성글고 이동성이 높은 경우에도 메쉬 유지를 위해 빈번하게 JOIN QUERY를 플러딩해야 하는 필요성이 감소하여 ODMRP에 비해 제어 메시지 발생도 줄어든다. 결국, PatchODMRP는 송신원의 수가 적고 이동성이 높은 환경에서 ODMRP에 비하여 JOIN QUERY 주기와 이동성 정도에 덜 민감하며 비교적 낮은 오버헤드로 높은 데이타 전송률을 유지할 수 있다.

먼저, PatchODMRP의 동작을 위해 변형된 라우팅 테이블과 포워딩 테이블의 형식에 대하여 간단하게 설명하고자 한다. [표 5]는 PatchODMRP를 위해 변형된 라우팅 테이블의 구조를 보여주고 있다. PatchODMRP에서의 라우팅 테이블은 기존의 ODMRP 라우팅 테이블 구조에 송신원으로부터의 홉 카운트 수를 기록하는 필드를 추가하였다. 호스트는 JOIN QUERY 패킷을 받았을 때, 라우팅 테이블에 JOIN QUERY의 송신원과 JOIN QUERY를 전달한 이전 홉의 주소뿐 아니라 송신원으로의 경유 카운트인 JOIN QUERY의 홉 카운트 필드 값을 기록한다. 한편, 포워딩 테이블은 [표 6]에서 보듯이 ODMRP에 비하여 상위 FG 노드의 IP 주소와 이 노드가 담당하는 송신원의 IP 주소를 리스트로 기록하는 필드를 추가하였다. ODMRP 포워딩 테이블 기존 필드들과 마찬가지로 JOIN REPLY를 전달하면서 이들 새로운 필드의 값을 수정하게 된다. 상위 FG 노드의 IP 주소 필드에는 JOIN REPLY의 NEXT_HOP_ID에 기록하는 값을, 이 노드가 담당하는 송신원의 IP 주소는 JOIN REPLY의 송신원 IP 주소를 저장한다.

표 5 라우팅 테이블 구성

목적지 (JOIN QUERY의 송신원) IP	다음 홉 (JOIN QUERY를 전달해준 전 홉) IP	홉 카운트

표 6 포워딩 테이블 구성

멀티캐스트 그룹 ID	타이머	상위 FG 노드의 IP 주소	송신원 IP 주소 리스트

PatchODMRP의 경로 손실 감지와 국부적인 경로 재설정에 관한 동작은 다음과 같다. 일반 에드-혹 호스트는 IEEE 802.11에서 제공하는 MAC계층의 BEACON 신호를[9] 이용하여, 현재 자신의 전송 범위 내에 어떤 이웃 호스트가 있는지를 파악할 수 있다. PatchODMRP에서는 이 정보를 이용하여 현재포워딩 테이블에 있는 상위 FG 노드들 중 전송 범위를 벗어난 것이 있는지 체크한다.

만약 그런 것이 있다면, FG 노드는 메쉬가 끊어졌을지도 모른다고 가정하고 새로운 데이타 경로를 복구하기 위한 작업을 시작한다. 이 작업은 ODMRP의 메쉬 형성 작업처럼, 요구 단계와 응답 단계로 진행된다. 먼저, **요구 단계**에서는 FG 노드가 인접한 FG 노드중 새 상위 FG 노드로 적합한 호스트를 찾기 위해 [표 7]과 같은 ADVT 패킷을 만들어 플러딩한다. 이 패킷은 상위 FG 노드를 잃어버린 FG 노드가 자신을 알리기 위한 것으로, 패킷 내의 {MG ID, SrcAddr, SrcHcount} 리스트는 포워딩 테이블과 라우팅 테이블을 통해 그 값들을 파악하여 기록하게 된다. MG ID와 SrcAddr은 각각 손실된 상위 FG 노드가 지원하던 멀티캐스트 그룹 ID와 멀티캐스트 송신원이며 이들 정보는 모두 포워딩 테이블에서 얻는다. SrcHcount에는 ADVT 패킷을 생성하는 FG 노드 자신(ADVTSrcAddr)으로부터 멀티캐스트 송신원까지의 홉 카운트를 기록하며 라우팅 테이블의 홉카운트를 참조하여 작성한다. 만일, 상위 호스트가 담당하던 멀티캐스트 그룹이 하나 이상이라면, 하나

표 7 ADVT 패킷 형식

{MG ID, SrcAddr, SrcHcount}리스트	ADVTSrc Addr	Prehop Addr	Hopcount

MG ID : 이전 상위 FG 노드가 지원해 주던 멀티캐스트 그룹 ID
SrcAddr : 이전 상위 FG 노드가 지원해 주던 멀티캐스트 그룹의 송신원 주소
SrcHcount : ADVTSrcAddr에서 송신원까지의 홉 카운트
ADVTSrcAddr : ADVT 패킷을 생성한 호스트 주소
PrehopAddr : ADVT 패킷을 전달해준 이전 홉 호스트의 주소(처음에는 ADVTSrcAddr와 동일)
Hopcount : ADVT의 라이프타임

의 ADVT 패킷에 리스트로 기록한다.

한편, ADVT 패킷 플러딩시, 플러딩 오버헤드를 줄이기 위해 ADVT의 최대 경유 홉 수를 2 ~ 3으로 제한하여 국부적으로만 플러딩되도록 한다. ADVT 패킷을 발생하는 FG 노드가 크게 멀티캐스트 지역을 벗어난 것이 아닌 경우(이 경우에만 해당 FG 노드가 메쉬에 연결되는 것이 필요함) 대부분 짧은 거리내에서 다른 FG 노드를 발견할 수 있을 것이라는 가정에 따라 불필요한 플러딩 오버헤드를 줄이기 위해 ADVT의 최대 경유 홉수를 이와 같이 짧게 설정하였다. 이 밖에, ADVT 패킷 필드에 포함해야 하는 정보는 [표 7]에서 상세히 설명하였다.

이 ADVT 패킷을 전송 받은 호스트는, 먼저 패킷의 중복을 검사한다. 중복 패킷이 아닌 경우, 호스트는 라우팅 테이블의 목적지 필드에 ADVT 패킷의 ADVTSrcAddr을 기록하고 다음 홉 필드에는 ADVT 패킷의 PrehopAddr을 기록하므로써 ADVT를 생성한 FG 노드로의 라우팅 정보를 갱신한다. 이것은 JOIN QUERY가 전달된 역경로를 기록해 두었다가 JOIN REPLY를 전달하는 경로로 사용한 것과 마찬가지로 ADVT 패킷이 지나간 역경로를 기록해 두었다가 PatchODMRP에서 ADVT패킷에 대한 응답으로 발생하는 제어 패킷인 PATCH 패킷이 전달될 때 그 루트로 사용하기 위함이다.

ADVT 패킷을 받은 호스트는 이렇게 라우팅 테이블을 업데이트 한 후, 다음의 세 가지 조건을 검사한다. 첫째, 자신의 포워딩 테이블에 있는 멀티캐스트 그룹 ID 중 ADVT에서 표시한 멀티캐스트 그룹 ID(MG ID)와 일치하는 것이 있는지, 둘째, 그 멀티캐스트 그룹에 대해 자신의 상위 FG 노드가 지원하는 송신원과 ADVT에서 요구하는 송신원(SrcAddr)이 같은지, 셋째, 그 멀티캐스트 송신원까지의 홉 수가 ADVT 패킷에 표시된 SrcHcount보다 작은지 검사한다. 세 번째 조건은 원래의 메쉬 형성 시에 ADVT를 생성한 FG 노드보다 해당 멀티캐스트 그룹의 송신원으로부터 더 가까웠던 FG 노드만이 ADVT를 생성한 FG 노드를 연결할 수 있도록 제한하기 위함이다. 이 조건이 모두 맞으면, 호스트는 자신이 ADVT를 생성한 FG 노드를 메쉬에 연결해 줄 수 있다고 판단하여 더 이상 ADVT를 전송하지 않고 이에 대한 응답단계에 들어가게 된다. 한편, 앞의 세 조건 중 하나라도 만족되지 않으면, 호스트는ADVT 패킷의 PrehopAddr에 자신 IP 주소를 적고 홉 카운트를 1 감소시켜, ADVT 패킷을 플러딩한다. 이 때 ADVT 패킷의 홉 카운트가 0이 되면 더 이상 전달하지 않는다.

응답단계에서는 ADVT 패킷에 대한 응답으로 PATCH 패킷을 만들어 ADVT를 발생한 FG 노드로 전송한다. PATCH 패킷의 구조는 [표 8]과 같다. 이 패킷도 ADVT 패킷과 마찬가지로 PatchODMRP에서 추가한 제어 패킷으로, ADVT 패킷에 대한 응답 패킷이다. PATCH 패킷의 ADVTSrcAddr 는 받은 ADVT 패킷의 ADVTSrcAddr를 복사한 것이고, PATCHAddr에는 PATCH 패킷을 생성하는 호스트의 주소가 들어가게 된다. NexthopAddr에는 ADVTSrcAddr로 가기 위한 다음 홉을 기록한다. PrehopAddr에는, 처음 PATCH 패킷이 생성될 때에는 PATCH 패킷을 생성하는 호스트의 주소를 기록하고, 그 후부터는 PATCH 패킷을 전달해 준 이전 홉 호스트의 주소를 적는다. 이렇게 만들어진 PATCH 패킷은 ADVT 패킷이 경유한 역경로를 통해 ADVT를 생성한 FG 노드(ADVT SrcAddr)에게로 전송된다. PATCH 패킷의 형식은 다음과 같다.

표 8 PATCH 패킷 형식

(MG id, SrcAddr, SrcHcount)list	PATCH Addr	ADVT SrcAddr	Nextho pAddr	Preho pAddr

MG id : ADVT 패킷의 멀티캐스트 그룹 ID
SrcAddr : ADVT 패킷에서 요구한 송신원 중, PATCH Addr가 지원하는 송신자 주소
SrcHcount : SrcAddr까지의 홉 카운트
PATCHAddr : PATCH 패킷을 생성한 호스트 주소
ADVTSrcAddr : ADVT 패킷을 생성한 호스트 주소 (ADVT와 동일)
NexthopAddr : PATCH 패킷이 전달될 다음 홉 주소(라우팅 테이블 참조)
PrehopAddr : PATCH 패킷을 전달한 이전 홉 주소

PATCH 패킷이 만들어지면, 라우팅 테이블이 제공하는 경로를 따라 ADVTSrcAddr까지 전달되게 된다. PATCH 패킷을 받은 호스트는 자신의 IP주소와 패킷의 NexthopAddr가 일치하면, 해당 멀티캐스트의 임시 FG 노드로 선출된다. 호스트는 먼저 자신의 라우팅 테이블의 목적지, 다음 홉, 홉 카운트 필드에 각각 PATCH 패킷의 SrcAddr, PrehopAddr, SrcHcount+1에 해당하는 값을 기록하고, 포워딩 테이블의 멀티캐스트 그룹 ID 필드와 상위 FG 노드의 IP주소 리스트 필드에 각각 패킷의 MG ID, PrehopAddr을 기록한다. 그리고 라우팅 테이블에서 ADVTSrcAddr로 가기 위한 다음 홉 주소를 찾아, PATCH의 NexthopAddr를 그 값으로 변경하고, SrcHcount는 1 증가시켜 ADVTSrcAddr를 향해 PATCH 패킷을 전송한다. PATCH 패킷을 받은 호스트의 IP 주소가 패킷의 NexthopAddr와 일치하지 않는다면, 그 호스트는 더 이상 PATCH 패킷을 전달하지 않고 버린다.

한편, PATCH 패킷을 받은 ADVTSrcAddr는 바로 새 경로를 결정하지 않고, 일정 시간동안 전송 받는 PATCH 패킷들을 [표 9]과 같은 구조를 가지는 PATCH 캐쉬에 모아둔다. 이는 ADVT를 발생시킨 호스트가 ADVT 패킷 발생 후부터 새 경로가 선택되기 이전까지 유지하는 캐쉬로써, 여러 개의 응답 PATCH 패킷이 있을 경우 이들이 제안하는 새로운 경로 중에서 멀티캐스트 그룹의 송신원과 가장 가까운 경로를 선별하기 위해 사용된다. ADVT패킷을 발생시킨 호스트는 PATCH 패킷을 받으면, 그 패킷의 {MG ID, SrcAddr, SrcHcount} 리스트 정보를 그대로 PATCH 캐쉬의 멀티캐스트 그룹 ID, 송신원, 홉 카운트 필드에 삽입하고, 다음 홉에는 PATCH 패킷을 자신에게 전달해 준 전 홉 IP주소를 기록한다. 이렇게 PATCH 캐쉬에 모은 메쉬 연결 경로 정보 중, SrcHcount가 가장 작은 엔트리의 정보가 선택된다.

표 9 PATCH 캐쉬 구조

멀티캐스트 그룹 ID	송신원	홉 카운트	다음 홉 (PATCH 패킷의 이전 홉) IP

[그림 1]은 위에서 설명한 PatchODMRP 동작의 예를 보여주고 있다.

[그림 1] (a)에서는 송신원 S, 수신원 R, FG 노드 I, J, K 로 ODMRP 메쉬가 형성되어 있는 상태를 보여준다. K의 포워딩 테이블에는 상위 FG 노드로 J를, 멀티캐스트 송신원으로 S를 표시해 두고 있다. 만일, J가 오

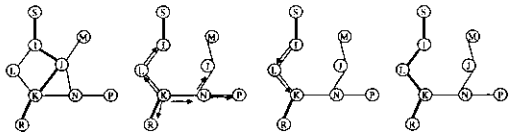


그림 1 PatchODMRP의 국부적인 플러딩으로 경로가 갱신되는 예

른쪽으로 이동을 하여 K와의 통신 범위를 벗어나 [그림 1](b)와 같이 되면, J와 K는 더 이상 BEACON 신호를 교환할 수 없다. K가 J로부터 BEACON 응답을 받지 못하여 연결이 단절되었음을 발견하게 되면, K는 J가 자신의 상위 FG 노드인지 포워딩 테이블에서 검사한다. 그 결과, 상위 FG 노드이므로, 송신원 S로의 데이타 경로 손실이 발생했을 수 있음을 인지하고, 이를 대처할 새 경로를 찾는 작업을 시작한다. K가 ADVT 패킷을 만들어 최대 전송 홉 수를 2로 하여 플러딩하면, [그림 1]의 (b)처럼 L, N, R 에게 우선 전달된다. 그러나 L, N은 FG 노드가 아니므로, L의 경우 라우팅 테이블의 목적지 IP 주소 필드와 다음 홉 주소 필드에 모두 K를 기록하고 플러딩한다. I가 L로부터 ADVT 패킷을 받으면, I는 먼저 ADVT테이블의 목적지 IP 주소 필드와 다음 홉 주소필드에 각각 K와 L을 기록한 후, 자신이 담당하는 멀티캐스트 그룹 송신원과 K에서 요구하는 멀티캐스트 그룹 송신원이 같은 지, 그리고 원래의 메쉬에서 송신원까지의 거리가 K보다 가까웠는지를 체크한다. 이 조건이 모두 맞으므로, I는 K에게 PATCH 패킷을 보낸다. [그림 1]의 (c)에서 PATCH 패킷은 I에서 K까지 중간 호스트를 모두 FG 노드로 선출하며, 라우팅 테이블에서 찾은 경로에 따라 L을 거쳐 K에 도착한다. K가 이 PATCH 패킷을 받으면, J대신 L을 상위 FG 노드로 채택함으로써, [그림 1]의 (d) 처럼 송신원 S로의 새로운 데이타 경로를 만든다.

4. 성능 평가

송신원의 수가 적은 경우 노드의 이동 속도가 증가함에 따라 ODMRP와 PatchODMRP의 성능이 어떻게 변화하는지 비교하기 위해 시뮬레이션을 수행하였다.

4.1 시뮬레이션 환경

시뮬레이션은 WindowsNT 4.0 / Microsoft Visual C++6.0 환경에서 Global Mobile Simulation (GloMo Sim) 라이브러리를 사용하여 구현되었다. 이 GloMo Sim 라이브러리는 UCLA에서 무선 네트워크 시뮬레이션을 위해 개발한 도구로서, 라이브러리 기반으로 순차적이고 병렬 처리가 가능한 시뮬레이터이다. GloMoSim

은 여러 개의 라이브러리 모듈로 구성되어 있으며, 이 모듈 각각은 특정 통신 프로토콜의 시뮬레이션을 수행한다. 이 라이브러리는 C 기반 병렬 시뮬레이션 언어인 PARSEC으로 개발되었다[10]. 따라서 새로운 프로토콜과 모듈들은 PARSEC을 사용하여 프로그램되거나 수정할 수 있다.

GloMoSim에서는 각 계층마다 기존에 연구된 여러 네트워크 프로토콜들이 라이브러리로 구현되어 있다. 또한, 에드-혹 네트워크처럼 호스트의 위치가 계속적으로 변하며, 라우팅이 여러 홉을 경유해야 하는 환경에 대해 계층별로 사용자정의에 따라 구현하도록 되어 있으며, MAC 계층과 네트워크 계층에서 제어 메시지를 교환하는 상세한 모델을 여러 가지 제시해 주고 있다. 표 10은 GloMoSim에서 제공하는 계층별 프로토콜을 보여준 것이다.

표 10 GloMoSim에서 제공하는 계층별 프로토콜

Layer:	Models:
Physical (Radio propagation)	Free space, Rayleigh, Ricean, SIRCIM
Data Link (MAC)	CSMA, MACA, MAC AW, FAMA, 802.11
Network (Routing)	Flooding, Bellman-Ford, OSPF, DSR, WRP
Transport	TCP, UDP
Application	Telnet, FTP

본 시뮬레이션에서는 Radio propagation 모델로는 Free space 전송을, MAC 프로토콜로써는 IEEE 802.11 Distributed Coordination Function(DCF)를 사용하였다. Free Space 모델은 채널 전파 계층(channel propagation layer)에 해당하는 프로토콜로써, 모든 송신원과 수신원사이의 거리만을 기반으로 신호 강도를 계산하는 모델이다. MAC계층의 IEEE 802.11은 Carrier Sense Multiple Access/Collision Avoidance(CSMA/CD)with acknowledgements를 사용한다. Radio type은 radio-capture로 하였으며, 네트워크 계층에서는 IP 프로토콜을 사용하였다. 응용 계층에서는 일정한 비트율로 데이타를 전송하는 CBR을 택하여 1초당 데이타 패킷을 하나씩 전송했다.

4.2 시뮬레이션 모델

시뮬레이션 모델은 다음과 같다. 우선, 1000m x1000m 지역에 호스트 50개를 무작위로 배치하였다. 그리고 이 호스트들 중에서 멀티캐스트 멤버를 다시 무작위로 선정한다. 이 멤버들은 시뮬레이션 시작 시 결정되어 끝날 때까지 멤버로 남아있게 된다. 멀티캐스트 그룹 멤버수는 최소 5에서 최대 10개까지 무작위로 결정되며 이중에 하나는 송신원이다. 송신원은 1초당 1개의 패킷을 내보내며, 그 페이로드 크기는 512bytes 이다. 각 호스트의 전파 전송 범위는 250 미터이며, 채널 용량은 2Mbits/sec 이다. 총 시뮬레이션 시간은 900초며, 초기 치를 바꿔가면서 여러 번 실행시켜 가능한 평균치를 뽑아내도록 하였다. 호스트의 움직임 모델로는 Random waypoint 를 사용하였다. 이 모델에 의하면 호스트는 이동할 목적지를 무작위로 선택하여 일정 속도로 그 목적지를 향해 움직여간다. 그리고 일단, 그 목적지에 도착하면, 호스트는 그 자리에서 일정 시간동안 머물러 있게 된다. 다시 일정 시간이 경과하면, 다시 호스트는 다른 방향의 목적지를 향해 위의 동작을 반복한다. 따라서 일정한 방향성 없이, 주어진 속도와 멈춤 시간에 의해서 호스트의 움직임이 발생한다. 시뮬레이션에서는 호스트의 속도를 다양하게 0 km/hr 에서 80km/hr 까지 실험하였다. ODMRP와 PatchODMRP의 오버헤드를 비교해 보기 위해서는 각 스킴에 따라 적합한 JOIN INTERVAL 값을 선택하고, 이 선택된 값을 사용하는 경우에 소요되는 오버헤드를 측정하였다.

ODMRP의 권고안에[4] 의하면 FG 노드의 LIFETIME값은 JOIN INTERVAL 값의 3 ~ 5배로 설정되어야 한다. 여기에서 LIFETIME이란 선출된 FG 노드가 해당 멀티캐스트 그룹의 FG 노드로서 작동하는 기간이다. 본 시뮬레이션에서는 ODMRP의 권고안[4]에 따라 ODMRP의 경우에는 FG 노드의 LIFETIME 값을 JOIN INTERVAL 값의 3~5배로 하였다. Patch ODMRP에서는 JOIN REPLY로 선출되는 일반 FG노드의 경우는 ODMRP와 같이 하였고, PATCH 패킷에 의해 임시로 선출되는 FG 노드의 LIFETIME값은 JOIN INTERVAL 값의 1/2배로 하였다. 이는, 임시로 선출된 FG 노드가 지나치게 많아져 중복적으로 데이타가 플러딩되는 것을 막기 위함이다.

한편, ODMRP와 PatchODMRP를 비교하기 위해 다음의 4가지 기준을 정하였다. 이중 몇 가지는 IETF MANET 워킹그룹에서 프로토콜 평가를 위해 제안한 기준이다[11].

가. 평균 데이타 패킷 전송률(A): 송신원에서 수신원으로 보낸 데이타 패킷이 평균적으로 얼마나 제대로 전달되었는지를 보여준다. 이 숫자는 프로토콜의 유효 정도를 나타낸다. 먼저 멀티캐스트 그룹의 각 수신원이 받은 데이타 패킷수(중복적으로 도착한 패킷은 하나만 셈)를 합하여, 이를 수신원수로 나눔으로써 평균적으로

하나의 수신원이 성공적으로 받은 데이타 패킷수를 계산하였다. 그리고 다시 이 값과 송신원이 보낸 총 데이타 패킷수의 비율을 구함으로써 평균 데이타 패킷 전송률을 계산하였다.

$$A=\frac{\frac{\sum \text{각수신원이받은데이타패킷수}}{\text{수신원수}}}{\text{송신원에서보낸총데이타패킷수}}$$

나. 수신원에 성공적으로 도착한 데이타 패킷 당 네트워크 전체에 발생한 제어 패킷의 수(B): 하나의 데이타 패킷을 전달하기 위해 발생되는 제어 패킷이 얼마나 되는지를 보여준다. 에드-혹 네트워크에서는 채널 용량의 한계 때문에 패킷의 수뿐 아니라 그 패킷들이 차지하는 바이트도 중요하다. 그러나, ODMRP와 PatchODMRP의 경우 기본 패킷 형식이 거의 유사하여, 그 크기에 차이가 거의 없으므로 패킷 수의 비율을 측정하였다. 이때, ODMRP의 경우 JOIN QUERY, JOIN REPLY를, PatchODMRP의 경우는 JOIN QUERY, JOIN REPLY, ADVT, PATCH 패킷을 측정하였다. 데이타 패킷당 발생한 제어 패킷 수는 시뮬레이션동안 네트워크 각 노드에서 해당 멀티캐스트를 위해 발생한 제어 패킷 수의 합을 구하여 이를 각 수신원에 성공적으로 전달된 데이타 패킷 수의 합으로 나눔으로써 계산하였다. 이 때, 성공적으로 전달된 데이타 패킷 수에는 역시 중복적으로 전달된 패킷은 포함시키지 않았다.

$$B=\frac{\text{총제어패킷수}}{\sum \text{각수신원에도착한데이타패킷수}}$$

다. 수신원에 성공적으로 도착한 데이타 패킷 당 네트워크 전체에 발생한 중복된 데이타 패킷의 수(C): 메쉬 구조에서는 데이타 패킷이 여러 경로를 통해 전달되어 결과적으로 수신원이 동일한 데이타 패킷을 중복적으로 받을 수 있다. 따라서 제어 패킷 이외에도 이러한 중복된 데이타 패킷이 채널 오버헤드가 될 수 있다. 이 오버헤드를 측정하기 위해, 시뮬레이션 동안 네트워크의 각 호스트들이 해당 멀티캐스트 그룹의 데이타 패킷 전달을 위해 발생시킨 데이타 패킷 수를 모두 더한 합을 구하고, 이를 수신원에 성공적으로 도착한 데이타 패킷 수(중복된 것은 제외)로 나눔으로써, 데이타 패킷 하나 당 발생된 데이타 오버헤드를 계산하였다.

$$C=\frac{\text{네트워크에발생한총데이타패킷수}}{\sum \text{각수신원에도착한데이타패킷수}}$$

라. 총 오버헤드(D): 네트워크 전체에서 해당 멀티캐스트를 위해 시뮬레이션 기간동안 발생된 모든 제어 패킷 수와 중복적으로 전달된 데이타 패킷 수의 합으로써 전체 오버헤드를 측정해 보았다.

$$D=\text{총제어패킷수}+\text{총데이타패킷수}$$

4.3 시뮬레이션 결과

▸ 평균 데이타 패킷 전송률

[그림 2]는 JOIN QUERY를 플러딩하는 인터벌인 JOIN INTERVAL을 3초, 60초, 120초, 240초 로 설정한 각 경우에 대하여 노드 이동성이 변화함에 따라 데이타 전송률(A)이 어떻게 변화하는지를 보여주고 있다. [그림 2]의 (a)에서, ODMRP의 경우에는 모든 JOIN INTERVAL 크기에 대해 호스트의 이동성이 커질수록 데이타 전송률이 감소함을 볼 수 있다. 이는 호스트의 이동성이 커질수록 메쉬 구성이 이를 따라가지 못해 경로 손실이 발생하는 경우가 많아지게 되고, 이에 의해 데이타의 손실이 늘어나기 때문이다. 또한, 호스트의 이동성이 같은 경우들을 비교해 보면, JOIN INTERVAL이 증가할수록 메쉬가 노드 이동에 대응해 빠르게 재구성되지 못해서 데이타 전송률이 감소함을 볼 수 있다. 반면, 경로 손실이 발생한 경우 비교적 짧은 BEACON 인터벌 내에(일반적으로 수 초) 손실된 경로를 국부적으로 재 설정하는 PatchODMRP 경우, [그림 2]의 (b)에서 보듯이, JOIN INTERVAL이 120초, 240초로 큰 경우에 전송률이 높고, 호스트의 이동 정도에 상관없이 전송률을 일정하게 유지한다. 그런데, 의외로 PatchODMRP의 경우에는 JOIN INTERVAL이 3초, 60초와 같이 작은 경우에 오히려 데이타 전송률이 낮고, 또 이 경우에는 호스트의 이동성이 증가함에 따라 데이타 전송률이 감소함을 볼 수 있다. 이것은 JOIN INTERVAL이 작은 경우에 JOIN QUERY 플러딩으로 발생하는 PatchODMRP의 제어 패킷량이 많아지고, 여기에 호스트의 이동성 정도가 커지게 되면 PatchODMRP의 경로 재설정 작업까지 빈번하게 일어나 더욱 제어 패킷 오버헤드가 증가하기 때문이다. 제어 패킷 오버헤드가 늘어나 패킷들의 채널 사용 경쟁이 높아지면 이로 인해 데이타 패킷의 손실 발생이 증가하게 된다. 따라서, PatchODMRP에서는 ODMRP와 달리, JOIN INTERVAL이 상당히 큰 경우 채널 사용 경쟁이 줄어들어 오히려 더 높은 데이타 전송률을 나타낼 수 있다. 본 시뮬레이션에서는 BEACON 인터벌을 3초로

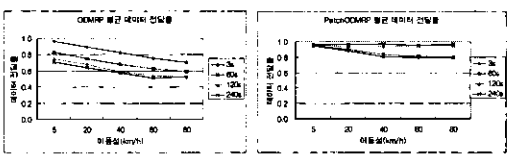


그림 2 평균 데이타 패킷 전달률

잡았기 때문에 JOIN INTERVAL이 3초인 경우에는 PatchODMRP의 메쉬 재구성 능력이 ODMRP의 메쉬 재구성 능력에 비해 향상되는 면이 거의 없기 때문에 [그림 2]의 (a)와 (b)를 비교해 보면 이 경우에서의 두 스킴의 데이타 전송률이 거의 비슷해짐을 알 수 있다.

위의 그림을 보면, JOIN QUERY주기가 ODMRP는 3초일 때, 그리고 pathODMRP는 240초일 경우 데이타 전송률이 우수함을 보인다. 따라서, 프로토콜 오버헤드 비교시 JOIN QUERY주기를 ODMRP는 3초, Patch ODMRP는 240초로 하였다. 이 경우의 데이타 패킷전송률은 PatchODMRP가 더 우수하다.

▸ 수신원에 성공적으로 도착한 데이타 패킷 당 발생하는 제어 패킷의 수

[그림 3]은 한 데이타 패킷당 발생한 제어 패킷 수(B)를 나타낸 것이다. 측정한 제어패킷의 종류에는 ODMRP의 경우 JOIN QUERY(JQ), JOIN REPLY(JR)이며, PatchODMRP에서는 JOIN QUERY(JQ), JOIN REPLY(JR), ADVT, PATCH이다. [그림 3]에 의하면 ODMRP의 경우에는 호스트 이동이 커질수록 데이타 패킷당 발생하는 제어 패킷 수가 증가한다. 제어 패킷 오버헤드의 주된 요인인 JOIN QUERY 플러딩은 JOIN INTERVAL마다 발생하므로 실제 발생하는 제어 패킷 수는 호스트 이동성 정도와 상관없이 거의 일정하지만, 호스트 속도가 60km 이상이 되면 수신원까지 성공적으로 도달하는 데이타 패킷 수가 현저히 줄어들어 도착 데이타 패킷당 발생하는 제어 패킷 수의 비율은 늘어난다. 이에 반하여, PatchODMRP에서는 이 값이 호스트의 속도가 증가하는 것에 별로 영향을 받지 않고 일정한 비율을 유지한다. 이는 도착한 데이타 패킷의 수가 호스트의 이동성 정도에 영향을 받지 않고 일정함을 의미한다. 그리고, 데이타 패킷 하나당 전송한 제어 패킷 비율도 ODMRP는 6~9인 반면 PatchODMRP는 1~2를 유지하여 제어 패킷 오버헤드가 훨씬 적음을 볼 수 있다. 한편, [그림 3]의 PatchODMRP에서 순수히 추가되는 제어 패킷인 ADVT와 PATCH이 차지하는 패킷량은 점선으로 표시하였다.

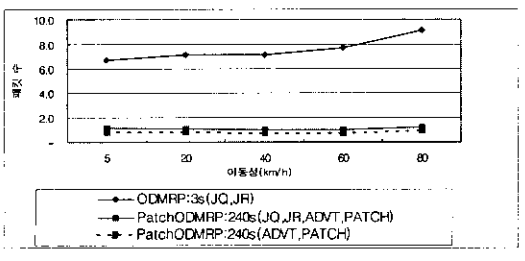


그림 3 데이타 패킷당 발생하는 제어 패킷의 수

▸ 수신원에 성공적으로 도착한 데이타 패킷당 발생한 중복된 데이타 패킷의 수

[그림 4]는 데이타 패킷 당 발생한 오버헤드 데이타 패킷 수(C)를 나타낸 것이다. 메쉬 구조에서는 하나의 데이타 패킷을 여러 경로로 전달하게 되기 때문에 제어 패킷에 의한 오버헤드 뿐아니라, 중복된 데이타를 전송하기 위한 오버헤드가 발생한다. [그림 4]에서 ODMRP와 PatchODMRP를 비교해 보았을 때, 둘 간의 데이타 패킷의 오버헤드는 큰 차이가 없다. 이를 통해 Patch ODMRP에서 국부적인 링크 단절을 고치기 위해 임시로 FG 노드를 선출하는 방식이 매우 효율적이어서, ODMRP에 비하여 필요 이상 지나치게 데이타 전달 경로가 중복적으로 발생하지 않음을 알 수 있다.

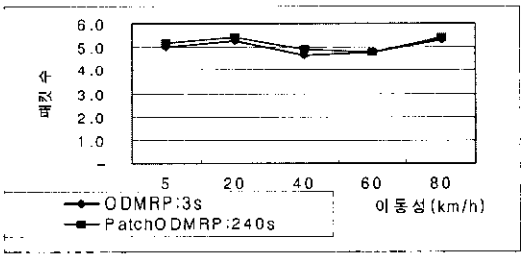


그림 4 수신원에서 성공적으로 도착한 데이타 패킷당 발생된 오버헤드 데이타 패킷의 수

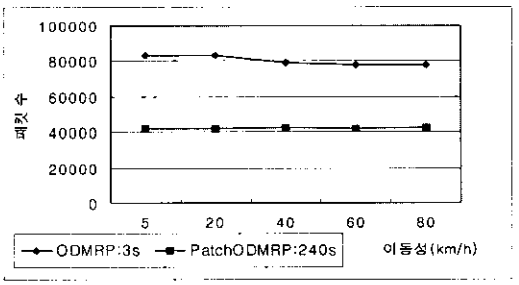


그림 5 네트워크 전체에 발생한 제어 패킷과 중복적으로 전달한 데이타 패킷

▸ 네트워크 전체에 발생한 총 오버헤드

에드-혹 네트워크 망에 발생한 전체 오버헤드 패킷의 양을 측정하기 위하여, 제어 패킷과 중복되어 전달된 데이타 패킷 수의 합을 구해보았다. [그림 5]를 보면,

PatchODMRP가 ODMRP의 약 1/2 배에 해당하는 오버헤드 패킷을 발생시키는 것을 알 수 있다. 이러한 비율은 호스트의 이동성이 증가하더라도 크게 변화가 없다. ODMRP의 오버헤드 양이 호스트 이동 정도가 증가할수록 약간 줄어든 이유는, 데이타 패킷이 수신원에 도착하지 못하고 중간에 손실되는 경우가 많이 발생하기 때문이다. 그에 반해 PatchODMRP에서는 일정한 수의 데이타 패킷이 계속 전달되었기 때문에 데이타 패킷 오버헤드가 이동성에 무관하게 일정 수준 유지되므로 전체 오버헤드가 이동성에 대해 거의 일정하게 유지되고 있다.

5. 결 론

본 논문에서는 멀티캐스트 그룹의 멤버수가 적고 특히 송신원의 수가 적어 ODMRP 메쉬가 성기게 형성되고, 노드의 이동성 정도가 높은 경우에 ODMRP의 성능을 향상시킬 수 있는 ODMRP 확장 방안으로 PatchODMRP를 제안하였다. PatchODMRP는 IEEE 802.11 의 BEACON신호를 활용하여 경로 손실을 감지하고 메쉬가 끊어졌을 염려가 있는 상황이면 국부적으로 메쉬 연결을 시도한다. ODMRP를 확장한 PatchODMRP의 성능 평가를 위해, 무선 네트워크 시뮬레이션에 적합한 GloMoSim 라이브러리를 사용하여 시뮬레이션을 수행하였다. 시뮬레이션에서는 ODMRP와 PatchODMRP의 성능을 평균 데이타 패킷 전송률, 수신원에 성공적으로 도착한 데이타 패킷과 제어 패킷 수의 비율, 수신원에 성공적으로 도착한 데이타 패킷과 중복된 데이타 패킷 수의 비율, 총 오버헤드 등의 4가지 기준으로 측정하였다. 시뮬레이션 결과, 송신원의 수가 적은 경우, 이동성이 높아질수록 PatchODMRP의 성능이 ODMRP의 성능보다 더욱 우수해지며, ODMRP에 비하여 적은 오버헤드로 높은 성능을 유지할 수 있음을 확인할 수 있었다.

참 고 문 헌

[1] E. Bommaiah, M.Liu, A. McAuley, and R. Talpade, "AMRoute:Ad-hoc Multicast Routing Protocol," Internet-Draft, draft-talpade-manet-amroute-00.txt, Aug. 1998.

[2] C.W. Wu, Y.C Tay, and C.-K. Toh, "Ad hoc Multicast Routing protocol utilizing Increasing id-numberS (AMRIS) Functional Specification," Internet-Draft, draft-ietf-manet-amris-spec-00.txt, Nov.1998.

[3] Sung-Ju Lee, Mario Gerla, Ching-Chuan Chiang, "On-Demand Multicast Routing Protocol," In Proceeding of IEEE WCNC'99. New Orleans, LA, Sep. 1999.

[4] Sung-Ju Lee, Willian Su, Mario Gerla, "On-Demand Multicast Routing Protocol (ODMRP) for Ad Hoc Networks," Internet Draft, draft-ietf-manet-odmrp-02.txt, July. 2000.

[5] Sung-Ju Lee, Willian Su, Mario Gerla, "Ad hoc Wireless Multicast with Mobility Prediction," In Proceeding of IEEE ICCCN'99, New Orleans, LA, Sep.1999.

[6] J.J. Garcia-Luna-Aceves, E.L. Madruga, "The Core-Assisted Mesh Protocol," IEEE Journal on Selected Areas in Communications, vol. 17, no.8, Aug. 1999.

[7] Sung-Ju Lee, William Su, Mario Gerla, and Rajive Bagrodia, "A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols," In Proceeding of Inforcom'2000.

[8] Ching-Chuan Chiang, Mario Gerla, L. Zhang, "Forwarding Group Multicast Protocol(FGMP) for Multihop, Mobile Wireless Networks," Baltzer Cluster Computing, vol.1, no.2, 1998.

[9] IEEE Computer Society LAN MAN Standards Committee, Wireless LAN Medium Access Protocol (MAC) and Physical Layer (PHY) Specification, IEEE Std 802.11-1997. The Institute of Electrical and Electronics Engineers, New York, 1997.

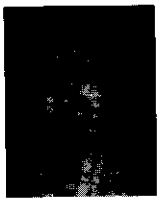
[10] UCLA Computer Science Department Parallel Computing Laboratory and Wireless Adaptive Mobility Laboratory, GloMoSim: A Scalable Simulation Environment for Wireless and Wired Network Systems. http://pcl.cs.ucla.edu/projects/domains/glomosim.html

[11] M.S. Corson and J. Macker, "Mobile ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," Request For Comments 2501, Internet Engineering Task Force, Jan. 1999.

[12] S.Corson, J. Macker, "Mobile Ad hoc Networking (MANET):Routing Protocol Performance Issues and Evaluation Considerations," RFC2501, Jan. 1999.

[13] J.Jubin and J. Tornow, "The DARPA Packet Radio Network Protocols," Proceeding of the IEEE, vol 75(1):21-32, 1987.

[14] Internet Engineering Task Force (IETF) Mobile Ad Hoc Networks (MANET) Working Group Charter, http://www.ietf.org/html.charters/manet-harter.html.



김 예 경

1992년 ~ 1996년 이화여자대학교 사회생활과 학사. 1995년 ~ 1997년 (주)두산정보통신 근무. 1998년 ~ 현재 이화여자대학교 컴퓨터학과 석사과정 졸업예정. 관심분야는 멀티캐스트 라우팅, 이동 데이타 통신.

이 미 정

1983년 ~ 1987년 이화여자대학교 전자계산학 학사. 1987년 ~ 1989년 University of North Carolina at Chapel Hill 컴퓨터학 석사. 1990년 ~ 1994년 North Carolina State University 컴퓨터공학 박사. 1994년 ~ 현재 이화여자대학교 공과대학 컴퓨터학과 부교수. 관심분야는 고속 통신 프로토콜 설계 및 성능 분석, 비디오 전송을 위한 트랙픽 제어, QoS 라우팅, 다중경로 라우팅, Ad-hoc 네트워크