애드 혹 유비쿼터스 컴퓨팅 환경을 위한 통합 라우팅 기법 연구

고양우 ° 이동만

정보통신대학교 공학부

newcat@icu.ac.kr, dlee@icu.ac.kr

SGR: A shared generic routing support for ad hoc ubiquitous computing environments

Yangwoo Ko° and Dongman Lee

Information and Communications University

요 약

애드 혹 유비쿼터스 컴퓨팅 환경의 가장 큰 특징은 연결의 역동성이다. 노드의 이동으로 인하여 두 노드를 연결하기 위한 경로를 지속적으로 재발견하여야 한다. 한편, 응용 계층에서도 사용자에게 현재의 상황에서 가장 적절한 서비스를 제공하기 위하여 연결된 서비스를 동적으로 재발견/재구성하게 된다. 이러한 환경에서는 이러한 재발견의 부담을 줄이는 것이 중요하다. 발견을 위한 비용이 응용 계층과 네트워크 계층에서 이중으로 발생하는 것을 회피하기 위하여 본 연구에서는 응용 계층의 레이블을 네트워크 계층에서 라우팅의 대상으로 사용하도록 제안한다. 물론 IP 주소에 기반한 라우팅도 동시에 지원한다. 본 연구에서는 IP 주소는 물론이고 응용 계층 레이블을 수용할 수 있는 공유된 라우팅 기법으로서 SGR(shared generic routing)을 제안한다. 시뮬레이션을 통하여 SGR이 발견의 비용을 효과적으로 줄일 수 있음을 보였고 리눅스 운영 체제 환경에서 구현하여 실제 적용 가능성을 제시하였다.

1. 서 론

애드 혹 유비쿼터스 컴퓨팅 환경에서는 응용이 그 환경에서 발견되는 서비스를 이용하여 작업을 수행한다. 기존의 서비스 발견 시스템에서는 각 서비스를 그 서비스가 운용되는 시스템의 IP 주소로 나타낸다. 인터넷에서와는 달리 이동형 애드 혹 네트워크(이하 MANET)에서는 어떤 IP 주소로 연결하기 위한 경로는 끊임없이 재발견되어야 한다. 따라서, 이러한 환경에서는 응용이 어떤 서비스를 활용하려면 이중의 발견 과정을 거쳐야 하며 이는 발견의 비용과 시간이라는 문제를 갖고 있다.

이중 발견의 부담을 해결하려는 여러 연구가 있었다. [1][2][3] 다른 계층의 정보를 활용함으로써 부담을 줄여보자는 계층간 접근이 전형적인 방식이다. M-ZRP[3]는 라우팅을 위한 컨트롤 메시지에 각 노드가 제공하는 서비스의 UUID를 껴 보냄으로써 부담을 줄인다. 하지만 이 방법은 각 서비스가 미리 정해진 고유 식별자로 구분되어야 하고 한 서비스는 한 노드에서만 제공한다는 등의 가정이 있어야만 동작하는 한계를 가진다. 또한 이와 같은 서비스 발견 방식은 일단 서비스 제공 노드가 발견되면 그 이후로는 그 노드의 IP 주소로만 통신을 하게 되므로 서비스의 상황 변화(예를 들어, 노드의 이동으로 더 가까이에 더 좋은 서비스가 나타난 경우)에 적응할 수 없게 된다. 즉, 응용 계층에서 생각하는 "가장 적절한" 노드를 네트워크 계층은 알 수가 없기 때문에 현재 연결된 노드와의 연결을 유지하기 위해 비용(예를 들어, 경로가 끊어진 경우의 복구 비용)을 쓰지만 응용 계층에서는 더 좋은 서비스와의 연결 기회를 잃어버리거나 혹은 더 좋은 서비스와 연결을 하게 된다면 네트워크 계층에서의 경로 복구 비용은 낭비되게 된다.

이중 발견의 부담을 근본적으로 제거하고 경로 발견에 있어 응용 계층의 요구를 반영하지 못하는 한계를 극복하기 위해서 본 연구에서는 IP 주소는 물론이고 응용 계층의 레이블을 라우팅의 대상으로 사용할 수 있는 SGR을 제시한다. SGR은 기존의 네트워크 계층의 하단부에 해당하는 부 계층으로서 그 위의 IP 라우팅이나 응용 계층 라우팅을 동시에 지원한다. MANET에서 경로를 발견하기 위해서 불가피하게 필요한 이웃 노드와의 토폴로지 정보 교환은 SGR에서 모두 담당하므로 그 위의 여러 라우팅 계층에서는 이러한 비용의 부담 없이 자신들이 필요한 라우팅을 실현할 수 있다. SGR 위에서 동작하는 간단한 유니캐스트, 멀티캐스트와 같은 IP 라우팅과 응용 계층 라우팅으로서 Publish/Subscribe(이하 pub/sub)를 구현하였으며 시뮬레이션을 통하여 더 적은 비용으로 더 높은 전송률을 구현할 수 있음을 보였다.

2. 기존 연구 분석 및 SGR과 특징 비교

M-ZRP[3]는 앞서 설명한 바와 같이 각 서비스에

UUID를 부여하고 이를 라우팅 메시지에 실어서 전파한다. UUID는 일반적으로 의미를 가지지 않으므로 부분적인 매칭과 같은 응용 계층의 요구에 따른 다양한 매칭을 지원할 수 없는데 반하여 응용 계층의 레이블을 지원하는 SGR은 이를 지원할 수 있다. M-ZRP에서는 일단 어떤 서비스를 지원하는 한 노드의 IP 주소를 알게 되면 그 이후로는 IP 주소만 사용하여 라우팅이 이뤄지는데 비하여 SGR에서는 각 홉에서 전송하는 과정에서까지 계속 응용 계층의 레이블을 사용하므로 전송 중에라도 상황에 따라 다른 곳으로 전송할 수 있다. 예를 들어, 어떤 데이터를 접근하려고 하는데 요청을 전송하는 중에 캐시 노드가 발견된다면 굳이 원래의 목적지까지 갈 필요가 없이 캐시 노드에게 전송하여 응답을 하게 할 수 있다. M-ZRP에서는 한 서비스를 한 노드만 제공한다고 가정하였지만 앞의 데이터 캐시의 예처럼 SGR에서는 이러한 제약을 두지 않음으로써 유연한 전송을 가능하게 할 뿐만 아니라 멀티캐스트, pub/sub와 같이 송수신자가 완전히 결부되지 않는 경우도 자연스럽게 수용할 수 있다.

[1]의 연구에서는 서비스를 찾기 위한 요청과 그 서비스로 접근하기 위한 요청을 하나의 요청으로 결합하였다. 어떤 노드가 어떤 서비스를 필요로 하면 그 서비스를 찾기 위한 요청을 라우팅 프로토콜의 메시지로서 발송을 하게 된다. 해당 서비스를 제공하는 노드의 주소를 알고 있는 노드가 요청을 받으면 그 요청을 그 주소로 접근하기 위한 경로 찾기 요청으로 바꾸어 전송한다. 즉, 한번의 요청으로 특정 서비스를 제공하는 노드로 가기 위한 경로까지 찾을 수 있게 된다. 앞의 M-ZRP 연구는 proactive 라우팅 기법에만 적용 가능하지만 이 연구는 reactive 라우팅에도 적용 가능하다. M-ZRP 연구와 마찬가지로 이 연구도 특정 서비스가 특정 노드에서만 제공되는 경우로 제약된다. 반면에 SGR은 한 서비스를 여러 노드가 지원하는 것이 가능하고 이는 각 수요자에 맞는 서비스 노드와 연결하는 것(예를 들어, 가장 가까운 캐시 노드로 연결)을 잘 지원할 수 있다.

앞의 두 연구는 라우팅 프로토콜이 응용 계층의 정보를 전달하는 도구로 사용된 반면에 어떤 연구에서는 네트워크 계층의 정보를 응용 계층이 활용하는 방안을 제시한다. 예를 들어, [2]의 연구는 MANET을 위한 pub/sub 기법을 제안하고 있다. 이 연구에서는 발행자와 구독자를 포괄하는 트리를 생성하고 발행자가 주기적으로 트리의 모든 노드에게 ping 메시지를 보내고 이 메시지는 지나온 경로를 기록한다. 메시지를 받은 노드는 트리 내에서의 자신의 위치와 메시지에 들어 있는 경로를 비교하여 네트워크 계층에서의 상대적인 위치가 트리 내에서의 위치와 맞는지를 확인하고 맞지 않으면 트리를 바꾸어 나간다. 이 연구는 다른 계층의 정보를 활용하여 더 빠른 적응을 할 수는 있으나 부담을 줄이지는 않는다.

SGR은 IP 주소 이외에도 응용 계층의 정보를 이용하여 라우팅을 구현하는데 이와 같이 IP 주소 이외의 정보를 이용하여 라우팅을 구현하는 연구가 여럿 있다. 첫째, 분산 해시 테이블을 이용하는 연구가 있다. I3(Internet Indirection Infrastructure)와 Tapestry가 대표적인 연구이다. [4][5] I3에서는 데이터를 보내는 것과 받는 것을 분리한다. 데이터를 보낼 때에는 특정 수신자에게 보내는 것이 아니라 어떤 식별자에게 보내고 이는 그 식별자를 담당하는 I3 서버에 전달된다. 데이터를 받고 싶으면 I3 서버에게 자신에 받고 싶은 식별자와 자신의 주소를 알려주면 I3 서버가 데이터를 전달해준다. I3 서버가 미리 배치되어 있는 것을 전제로 하고 있으므로 본 연구의 애드 혹 환경과는 맞지 않다. 또한, 모든 데이터가 I3 서버를 통하여 전송되므로 전송 경로가 최적이 아니고 I3 서버가 병목이 될 수 있다. IP 주소를 사용하지 않는 둘째 부류의 연구는 콘텐트 기반 라우팅 연구이다. [6][7] 예를 들어, [6]의 연구에서는 어떤 노드가 주소를 갖지도 않고 데이터를 특정 주소로 보내지도 않는다. 대신 각 노드는 자신이 어떤 콘텐트를 받고 싶은지를 나타내는 술어를 광고하고 그 술어와 매치되는 정보를 전송 받게 된다.

3. 설계 요구 사항

기존의 MANET 라우팅 프로토콜은 다음과 같은 특성을 공통적으로 갖고 있다: (1) 이웃 노드와의 정보를 공유하고 이를 통하여 정보를 전파하며 (2) 네트워크 정보는 soft state로 유지되며 실제로 사용되는 정보만 계속 유지된다. MANET의 공통 라우팅 기능을 하는 SGR은 다음과 같은 기능을 갖추어야 한다.

- 이웃 노드와 정보 교환: 이는 주기적으로 또는 즉시적으로 일어난다.
- 교환된 정보의 관리: 이웃과의 정보 교환을 통하여 입수한 정보는 필요한 그리고 가장 좋은 정보를 유지한다. 여기서 "가장 좋은"의 의미는 상위 라우팅 프로토콜의 요구에 따라 다를 수 있다. 예를 들어, 멀티캐스트의 경우 가장 가까이 있는 노드들끼리 연결하는 것이 바람직하지만 pub/sub의 경우에는 유사한 관심사를 가진 노드들끼리 연결하는 것이 더 바람직하다. [8]
- 경로의 유지와 폐기: 노드의 이동성 등으로 인하여 경로는 계속 변할 수 있으므로 현재 사용되고 잘 연결된 경로는 유지하고 아닌 경로는 복구 또는 폐기하여야 한다.

4. 제안 스킴 - SGR

4.1. 아키텍처

그림1에서 제시하는 바와 같이 SGR은 MAC 계층

위에서 동작하며 그 위로 여러 라우팅 프로토콜(이들을 SGR과 구분하기 위하여 "상위 라우팅 프로토콜"이라고 부른다.)을 지원한다. 그림을 간단히 하기 위해서 하나의 유니캐스트/멀티캐스트 상위 라우팅 프로토콜만 있는 것으로 그렸지만 필요하다면 여러 개의 유니캐스트/멀티캐스트 라우팅 프로토콜이 공존할 수도 있다. 또한, 특정 목적에 맞게 설계된 환경에서 IP 라우팅이 필요하지 않는다면 아예 이들을 배제하고 SGR과 특정 응용에 필요한 몇 개의 상위 라우팅 프로토콜만으로 구성하는 것도 가능하다.

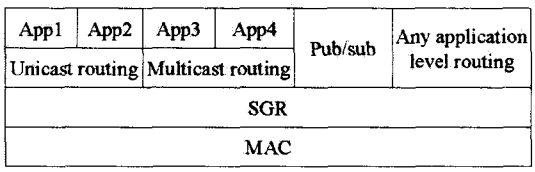

그림 1. SGR의 위치

여러 상위 라우팅 프로토콜이 공존하므로 어떤 수신 패킷을 어느 상위 라우팅 프로토콜이 다룰 것인지를 SGR 헤더에 기록하여야 하는데 이 필드를 TYPE이라고 부르기로 한다. 각 TYPE에는 다시 여러 라우팅 대상이 존재할 수 있다. 예를 들어, IP 라우팅에서의 IP 주소, pub/sub에서의 토픽 등이 이에 해당한다. 이러한 대상을 TAG라고 부르며 SGR 라우팅 테이블에서 (TYPE,TAG)가 키로 사용된다. 물론 하나의 TAG를 향하는 세션이 여럿이 공존할 수 있다. 예를 들어, 같은 IP 주소로 연결된 여러 TCP 세션이 있을 수 있으나 이들을 구분하는 것은 각 상위 라우팅 프로토콜의 역할이므로 (즉, IP 라우팅의 경우 포트 번호를 IP 헤더에 포함하므로) SGR에서는 관여할 필요가 없다.

SGR 라우팅 테이블에 엔트리가 추가되는 데에는 크게 나누어 그 노드 스스로 생성하는 경우와 이웃 노드와의 정보 공유를 통하여 알게 되는 경우 두 가지가 있다. 전자의 경우는 다시 "필요할 때" 추가하는 경우와 "미리" 추가 하는 경우가 있다. 이는 각각 reactive 라우팅과 proactive 라우팅에 사용된다. 즉, reactive IP 라우팅이 사용된다면 데이터그램이 어떤 IP 주소로 발송이 되려는데 해당 주소가 라우팅 테이블에 없는 경우 SGR에 (TYPE=reactive IP routing, TAG=해당 IP 주소)를 키로 하는 엔트리가 생성된다. 이때 키 필드를 제외한 다른 필드 값은 특별한 값으로 채워서 그 엔트리의 값이 실제 경로를 의미하는 것이 아니라 해당 경로를 아직 찾지 못하였다는 것을 의미하게 한다. 이 엔트리는 이웃 노드와의 교환을 통하여 네트워크에 전파되고 해당 IP 주소로 가는 경로를 아는 노드(예를 들어, 해당 IP 주소의 노드)가 갱신해서 다시 전파함으로써 경로를 노드들이 알게 된다. 한편, proactive 라우팅이 사용되는 경우에는 각 노드가 자기 자신으로 오기 위한 경로를 나타내는 엔트리를 자기의 SGR 라우팅 테이블에 미리 넣어둔다. 예를 들어, "오늘의 뉴스"라는 토픽을 제공하는 발행자라면 (TYPE=pub/sub, TAG="오늘의 뉴스")라는 키의 엔트리를 생성하고 홉 수 필드를 0으로 한다. 이 엔트리는 이웃과 공유되면서 네트워크로 전파되고 해당 토픽에 관심이 있는 구독자들이 어떤 경로로 발행자로 접근할 수 있는지 알게 된다.

이때까지 설명한 라우팅 테이블을 SGR에서는 CACHE 테이블이라 부르며 이 외에도 두 개의 테이블을 더 갖고 있다. RECEPTION 테이블은 자신이 수신해야 할 하는 TAG 목록(예를 들어, IP 유니캐스트 라우팅의 경우 자기 자신의 주소 또는 멀티캐스트나 pub/sub의 경우 자신이 가입한 그룹 또는 토픽)을 가지고 있고 FORWARD 테이블은 자신이 재전송해야 할 TAG의 목록을 갖고 있다. 즉, 자신이 어떤 TAG로 향한 "경로 위에" 있으면 해당 TAG가 FORWARD 테이블에 들어 있게 된다. CACHE 테이블은 다음과 같은 필드를 가진다.

- TYPE: 어느 상위 라우팅 프로토콜의 엔트리인지 나타낸다.
- TAG: 라우팅의 대상을 나타낸다.
- 다음 홉: 해당 TAG로 가기 위해 어느 홉을 거쳐서 가야 하는지 나타낸다.
- 시간 스탬프: 엔트리가 테이블에 추가된 시점을 나타낸다.
- 홉 수: 해당 TAG로 도달하기 위한 거리를 홉의 수로 나타낸다.

4.2. 동작 방식

SGR의 기본 동작 방식은 AODV와 유사하다. 즉, 필요한 경로를 찾기 위해서 경로를 찾는 패킷이 flood되고 이의 역경로로 정보가 전송되어 옴으로써 경로가 형성된다. 단, 경로를 찾기 위해 별도의 컨트롤 패킷을 보내는 대신 자신의 CACHE 테이블에 경로 찾기 요청에 해당하는 빈 엔트리를 추가함으로써 CACHE의 공유와 경로 찾기가 동시에 일어나게 된다. SGR이 하는 일은 주기적으로 하는 일과 특정 종류의 패킷을 받았을 때 하는 일도 나누어 볼 수 있다. 주기적으로 하는 일은 다음과 같다.

- 이웃 노드에게 CACHE 테이블을 브로캐스트한다. 이를 CA(Cache Advertisement)라고 부른다.
- 최근에 CA를 받은 적이 없으면 CA 요청(CA Solicitation)을 보낸다. 이 요청은 CA의 손실을 복구하는 기능과 주변에 이웃 노드가 하나도 없음을 알아내는 두 가지 기능을 한다.
- CACHE와 FORWARD 테이블의 오래된 엔트리를 삭제한다. 한편, RECEPTION 테이블의 엔트리는 상위 라우팅 프로토콜에서 명시적으로 삭제할 때까지 유지한다.

- FORWARD와 RECEPTION 테이블에 있는 엔트리 각각에 대하여 다음 홉 노드에게 JOIN 메시지를 보냄으로써 다음 홉 노드의 FORWARD 테이블에서 자신의 엔트리를 갱신해준다.

한편 특정 종류의 패킷을 받았을 때 SGR은 다음과 같은 일을 한다.

- CA Solicitation을 받으면 즉시 CA를 재전송한다.
- CA를 받으면 포함된 내용을 이용하여 자신의 CACHE 테이브를 갱신한다.
- JOIN을 받으면 자신의 FORWARD 테이블을 갱신한다. 만약, JOIN의 대상 TAG가 내가 아니라면 (즉, 내가 "경로 위의" 노드라면) 해당 TAG의 다음 홉 노드에게 JOIN을 전송한다.

상위 라우팅 프로토콜이 SGR을 활용할 수 있도록 SGR은 다음과 같은 API를 제공한다.

- TYPE 등록: 상위 라우팅 프로토콜이 초기화될 때 자기 자신을 SGR에 등록하는데 사용한다.
- TAG 등록: 자기 자신을 나타내는 TAG 또는 접속하려는 상대를 등록할 때 사용한다.
- 전송: SGR을 통해서 데이터 패킷을 전송한다.
- RECEPTION TAG 등록: 어떤 TAG로 오는 데이터 패킷을 받고 싶은지 등록한다.

한편 상위 라우팅 프로토콜은 SGR과 협력하기 위하여 다음과 같은 함수를 제공한다.

- 데이터 패킷 수신: SGR이 데이터 패킷을 수신하면 TYPE 필드를 이용하여 해당 상위 라우팅 프로토콜에 전달할 때 이 함수를 사용한다.
- 라우팅 테이블 갱신: CA를 통하여 이웃 노드로부터 받은 정보를 이용하여 SGR은 자신의 CACHE 테이블을 갱신한다. 그런데 상위 라우팅 프로토콜의 정책에 따라 갱신 방식이 달라질 수 있다. 앞에서 예시한 바와 같이 멀티캐스트 라우팅의 경우에는 단지 홉 수가 작은 것을 고르는 것이 데이터 전송의 효율을 높이지만 pub/sub의 경우에는 더 많은 토픽을 공유하는 이웃과 전송 경로를 형성하는 것이 더 효율적이다. 이렇듯 상위 라우팅 프로토콜 마다 다른 갱신 정책을 반영하기 위하여 이 갱신 함수를 호출한다. 이 함수가 등록되지 않은 경우에는 SGR의 기본 갱신 기능을 이용한다.

대부분의 라우팅 프로토콜은 데이터 전송 과정에서 손상된 경로가 발견되면 이를 뛰어 넘어서 전송하기 위해서 해당 패킷을 브로드캐스트하고 이를 받은 노드 중 경로를 가진 노드가 재전송하도록 한다. 이런 방식은 전송에 있어 오버헤드를 증가시키지만 전송률을 높이는 장점이 있다. 기존 라우팅 프로토콜과의 비교 편의를 위하여 SGR도 유사한 기능을 추가하였다. 한편, 오버헤드를 줄이기 위하여 브로드캐스트된 패킷을 누군가 전송하면 이웃 노드가 이를 엿듣고 자신의 전송을 억제하는 기법을 쓰는 경우도 있으나 이 연구의 목적이 더 좋은 라우팅 프로토콜을 만드는데 있는 것이 아니므로 이러한 추가 기능은 구현하지 않았다.

5. 성능 평가

SGR의 성능을 평가하기 위하여 ns-2 시뮬레이터를 이용하여 기존의 MANET 라우팅 프로토콜과 비교하였다. [9] 또한 실제 적용 가능성을 확인하기 위하여 SGR를 리눅스 운영 체제에 구현하였다.

기존의 MANET 라우팅과 비교하기 위하여 아주 간단한 reactive 유니캐스트 라우팅과 proactive 유니캐스트 라우팅을 각각 구현하였다. 응용 계층 라우팅과의 결합 가능성을 확인 하기 위하여 [8]에서 제시한 기법을 단순화하여 pub/sub 스킴을 구현하였다.

비교 대상이 되는 기존의 MANET 라우팅 프로토콜로는 AODV와 DSDV를 사용하였다. [10][11] 세 개의 유니캐스트 세션과 두 개의 pub/sub 세션을 동시에 실행하고 이때의 전송률과 오버헤드를 측정하였다. 한 pub/sub 세션에는 한 발행자에 두 구독자가 수신하는 것으로 하였다. 1000x1000m 공간에 50개의 노드가 있으며 이동성은 낮은 경우(최대 속도 1m/sec에 60초의 정지시간), 중간 경우(최대 속도 2m/sec에 30초의 정지시간), 높은 경우(최대 속도 3m/sec에 정지 없음)의 세 가지로 바꾸어 가며 실험하였다.

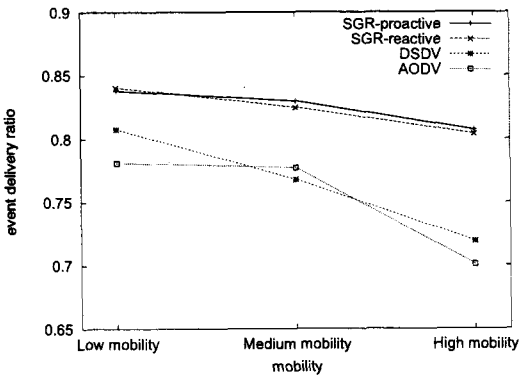

그림 2. Pub/Sub 세션의 패킷 전송률

그림2는 pub/sub 세션의 패킷 전송률을 보여주는데 SGR위에서 돌아가는 pub/sub가 기존의 MANET 프로토콜 기반의 구현보다 더 높은 전송률을 보이고 있다. 현재 구현된 pub/sub에서 전송 트리에 참여하기 위해서는 이미 트리에 포함된 노드에게 유니캐스트로 JOIN 메시지를 보낸다. SGR의 경우에는 유니캐스트 전송을 위해 생성된 라우팅 정보를 활용하여 pub/sub도 송수신을 하게 되므로 pub/sub를 위한 경로를 만드는 부담이 없고 유니캐스트로 전송이

가능했다면 pub/sub로도 전송이 가능할 확률이 높아진다. 또 다른 이유로는 SGR에서는 경로가 끊어진 경우 브로드캐스트로 전송할 때 억제를 하지 않으므로 전송률이 높다. 물론 억제가 없어서 오버헤드는 커지지만 SGR이 기존의 별개로 동작하는 라우팅 프로토콜보다 오버헤드가 충분히 적으므로 문제가 되지 않을 것이며 이는 아래의 오버헤드 평가에서 확인된다.

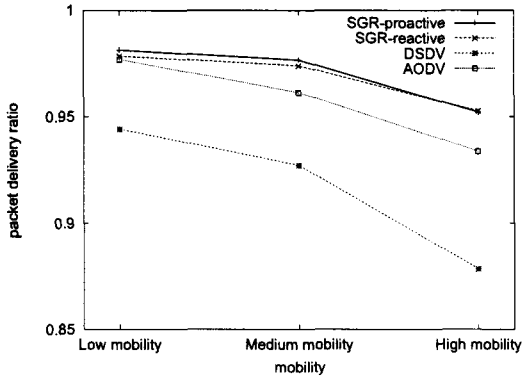

그림 3. 유니캐스트 세션의 패킷 전송률

그림3은 유니캐스트 세션의 패킷 전송률을 보여주고 있다. 그림2와 마찬가지로 기존의 MANET 프로토콜보다 SGR이 더 좋은 성능을 내긴 하지만 그 차이는 상대적으로 적고 전체적으로 pub/sub의 경우보다 높은 전송률을 보이고 있다. SGR기반의 유니캐스트 라우팅 프로토콜이 무척 단순함에도 불구하고 기존의 정교한 라우팅 프로토콜보다 더 좋은 성능을 낼 수 있는 것은 오버헤드를 더 쓰는 경로 복구 방식 때문이다. 공간의 제약으로 포함하지 않았지만 경로 복구 기능을 포함하지 않은 SGR의 경우에는 유니캐스트만 비교하면 기존의 MANET 라우팅 프로토콜보다 낮은 전송률을 보이는 반면 아주 낮은 오버헤드를 나타내었다.

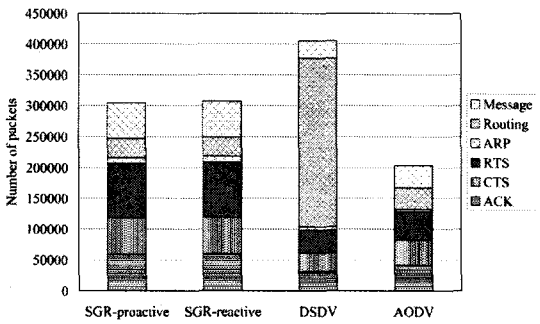

그림 4. MAC 계층에서 전송한 패킷의 수(유니캐스트 세션만 있는 경우)

그림 4와 5를 보면 SGR이 가진 특징이 아주 잘 드러난다. 그림 4는 앞의 시나리오와는 달리 유니캐스트 세션만 있는 경우의 오버헤드를 나타낸 것이다. 그림에서 보듯이 SGR은 DSDV보다는 적지만 AODV보다는 높은 부담을 보이며 이는 대부분 경로 복구 방식의 높은 부담 때문이다. 이는 SGR 위에서 구현한 유니캐스트 라우팅이 기존의 MANET 라우팅 프로토콜처럼 정교한 기능을 갖고 있지 않기 때문이다. 한편, 그림5는 원래의 시나리오대로 여러 종류의 세션이 섞여 있는 경우이며 AODV의 오버헤드는 급격히 늘어나서 SGR의 부담보다 더 커진다. 만약, 더욱 다양한 응용 계층 라우팅이 있다면 이러한 차이는 점점 더 커질 것이다. 이를 통해서 알 수 있듯이 SGR은 여러 종류의 상위 라우팅 요구가 공존하는 환경에서 통합된 라우팅 서비스를 제공함으로써 부담을 줄이고 전송률을 높이는 역할을 하게 된다.

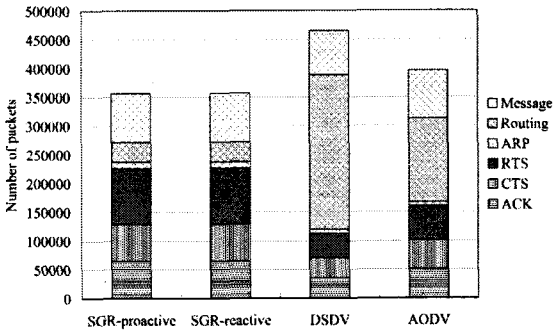

그림 5. MAC 계층에서 전송한 패킷의 수(pub/sub와 유니캐스트 세션이 같이 있는 경우)

실제 적용 가능성을 확인하기 위해서 리눅스 운영체제 위에서 SGR를 구현하였다. 경로가 없는 패킷의 송신을 가로채어 라우팅 프로토콜을 구동하기 위해서는 ASL을 이용하였다. [12] SGR를 통하여 확보한 경로를 운영체제의 라우팅 정보에 반영하기 위해서는 사용자 공간 명령어를 사용하였다. 시뮬레이션에서와 마찬가지로 SGR 위에서 동작할 수 있는 상위 라우팅 프로토콜로서 시뮬레이션에서와 같은 proactive IP 유니캐스트 라우팅과 pub/sub를 구현하였다.

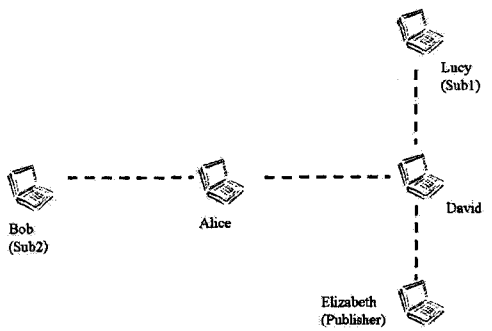

그림 6. MANET 에뮬레이션 시나리오

많은 노드와 이동성을 고려한 실험이 바람직하지만 현실적으로 힘들기 때문에 그림 6에서와 같이 배치하고 노드가 움직이지 않는 상황에서 실험하였다. 각 장비에는 하나의 WiFi 인터페이스가 장착되어 있고 이들은 IEEE 802.11b에 Ad Hoc 모드로 동작한다. 그림에서 점선은 한 홉으로 연결될 수 있는 이웃을 나타낸다. 서로 다른 세 개의 세션을 동시에 실행시켰는데 Bob은 Elizabeth에게 주기적으로 PING 메시지를 보내고 Lucy는 Bob에게 7MB의 파일을 SCP(secure copy)명령을 이용하여 전송하고 Elizabeth은 발행자가 되어 주기적으로 이벤트를 발송하고 Bob과 Lucy는 구독자로서 수신하도록 하였다.

PING 메시지의 RTT는 작게는 3.35ms부터 크게는 337.4ms까지 나타났다. 이와 같은 큰 변이가 나타나는 이유는 경로를 설정하는 과정에서 사용자 공간 명령을 사용하였고 이 명령의 실행은 운영체제의 프로세스 스케쥴링에 영향을 받기 때문으로 추정된다. 세 홉 거리에서의 SCP 성능은 27.9 ~ 35.8KB/sec을 보였다. WiFi의 성능과 여러 홉을 거치는 상황을 고려할 때 매우 높은 성능이라 할 수 있다. Bob에서 측정한 pub/sub 세션의 이벤트 전송률은 0.79~0.88인 반면에 Lucy에서 측정한 전송률은 0.93~0.99으로 나타나서 여러 홉을 거치는 경우 전송률이 기하급수적으로 떨어진다는 일반적인 결론과 일치함을 알 수 있었다.

6. 결론

상황에 맞춰 최적의 서비스가 지속적으로 변할 수 있는 애드 혹 유비쿼터스 컴퓨팅 환경에서는 통신의 대상을 유연하게 바꾸는 기능이 필수적이다. 따라서, 최적의 서비스를 네트워크에서 찾아내는 노력이 필요하다. 한편, MANET에서는 노드 간의 경로가 지속적으로 바뀌므로 경로의 변화도 지속적으로 감시하여야 한다. 따라서, 상황의 변화에 따라 새로운 서비스를 찾고 또한 그 서비스로 연결하기 위한 경로를 찾는 것은 이중의 부담이 될 뿐만 아니라 응용 계층에서의 상황 변화가 네트워크 계층에 곧바로 반영되지 않음으로 인한 경로 복구 비용의 낭비, 최적이 아닌 서비스로의 연결 등 부작용이 발생한다.

이러한 부담과 부작용을 근본적으로 해결하기 위하여 본 연구에서는 응용 계층의 요구 사항을 곧바로 라우팅 프로토콜의 접속 대상으로 표현할 수 있고 IP 라우팅은 물론이고 다양한 응용 계층 라우팅의 공통된 기능을 한곳에서 처리하는 공통 라우팅 기능으로 SGR을 제시한다. 시뮬레이션을 통하여 여러 종류의 세션이 공존하는 경우 SGR은 기존의 MANET 라우팅보다 더 높은 전송률과 더 낮은 부담을 보이는 것을 확인할 수 있었다.

Acknowledgement

본 연구는 21 세기 프론티어 연구개발사업의 일환으로 추진되고 있는 정보통신부의 유비쿼터스 컴퓨팅 및 네트워크 원천기반 기술개발사업의 지원에 의한 것임

참고문헌

[1] R. Koodli and C. E. Perkins. Service discovery in on-demand ad hoc networks, Oct. 2002. INTERNET-DRAFT draft-koodli-manet-servicediscovery-00.txt.

[2] A. V. Marco Avvenuti and G. Turi. A Cross-Layer Approach for Publish/Subscribe in Mobile Ad Hoc Networks. In Mobility Aware Technologies and Applications, volume 3744/2005 of LNCS, pages 203{214. Springer Berlin/Heidelberg, 2005.

[3] C. N. Ververidis and G. C. Polyzos. Routing layer support for service discovery in mobile ad hoc networks. In PERCOMW '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops, pages 258{262, Washington, DC, USA, 2005. IEEE Computer Society.

[4] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet indirection infrastructure, 2002.

[5] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, UC Berkeley, Apr. 2001.

[6] A. Carzaniga and A. Wolf. Content-based networking: A new communication infrastructure, 2001.

[7] M. Petrovic, V. Muthusamy, and H.-A. Jacobsen. Content-based routing in mobile ad hoc networks. In MOBIQUITOUS '05: Proceedings of the The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, pages 45{55, Washington, DC, USA, 2005. IEEE Computer Society.

[8] Y. Huang and H. Garcia-Molina. Publish/subscribe in a mobile environment. Wirel. Netw., 2004.

[9] The Network Simulator ns-2 (v2.1b8a). http://www.isi.edu/nsnam/ns/, October 2001.

[10] C. Perkins and E. M. Belding-Royer. Ad-hoc on-demand distance vector routing. pages 90{100, 1999.

[11] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications, 1994.

[12] V. Kawadia, Y. Zhang, and B. Gupta. System services for implementing ad hoc routing protocols, 2002.