MANETs 에서 안정적이고 효율적인 통신을 위한 클러스터 기반 분산 라우팅 프로토콜

장준혁[O1] 정진만[1] 민홍[1] 조유근[1] 김봉찬[2] 홍지만[2]
서울대학교 컴퓨터공학부[1]
숭실대학교 컴퓨터학부[2]
{jhjang[O], jmjung, hmin, cho}@os.snu.ac.kr[1], {bcgim, jiman}@ssu.ac.kr[2]

Stable and Efficient Distributed Cluster-based Routing Protocol for MANETs

Joonhyouk Jang[O1] Jinman Jung[1] Hong Min[1] Yookun Cho[1] Bongchan Gim[2] Jiman Hong[2]
School of Computer Science and Engineering, Seoul National University[1]
School of Computing, Soongsil University[2]
{jhjang[O], jmjung, hmin, cho}@os.snu.ac.kr[1], {bcgim, jiman}@ssu.ac.kr[2]

요 약

기존의 클러스터 기반 라우팅 프로토콜들은 클러스터 외부와의 통신을 클러스터 헤드가 전담함으로써 통신 경로의 구성을 단순화하고 클러스터 헤드를 통한 데이터 통신을 가능하게 하였다. 하지만 매 라운드마다 클러스터 헤드의 재선출(re-electioin) 및 클러스터의 재구성(re-clustering)에 드는 비용을 감수해야 한다. 본 논문에서는 클러스터 헤드없이 완전히 분산된 방식으로 클러스터를 구성하여 클러스터 헤드의 재선출 및 클러스터 재구성의 비용을 줄이고, 안정적인 클러스터 구조를 바탕으로 노드 간의 P2P 통신을 지원하는 라우팅 프로토콜(Distributed Cluster-based Routing Protocol, DCRP)을 제안한다. 제안 기법은 각각의 노드가 이웃 노드와의 연결 상태를 감지하여 독립적으로 자신의 상태를 변화시킴으로써 클러스터를 구성한다. 특정 클러스터를 지나는 경로는 완전히 해당 클러스터 내의 노드들의 책임 하에서 관리되기 때문에 에너지 소모를 효율적으로 관리하고 네트워크의 크기에 따른 확장성을 보장할 수 있다.

제안 기법은 시뮬레이션을 통해 기존 기법에 비해 네트워크 구조(topology)의 안정성이 향상되고 제어 메시지의 교환과 경로 설정에 소모되는 비용이 크게 줄었음을 확인하였다.

1. 서 론

무선 센서 네트워크는 온도, 소리, 진동, 압력, 움직임 등 각종 물리적인 상태를 감지하는 센서들로 구성된다. 그 중 모바일 애드혹 네트워크(MANETs)들은 유선 환경에 기반을 둔 AP 혹은 기지국 없이 노드들이 독립적으로 네트워크를 구축하여 통신을 주고받는 형태의 무선 네트워크를 말하며 중앙 집중식의 관리가 필요 없고 기반 시설이 없는 환경에서도 사용 가능하다는 장점이 있다. 이는 본래 전쟁 상황의 감시와 같은 군사적인 응용 목적으로 시작되었으나 최근에 이르러서는 환경, 헬스케어, 홈오토메이션, 교통 통제 등 여러 영역에서의 응용이 크게 늘어나는 추세다.[1][2][3]

하나의 노드는 무선 통신 장치, 마이크로 컨트롤러, 배터리로 이루어지며 에너지량과 메모리 크기 등 자원의 제약은 노드의 크기와 가격에 따라 달라진다. 이중 가장 큰 제약은 에너지이다. 일반적으로 한번 배치된 노드의 배터리를 재충전하는 경우는 거의 없기 때문에 각 노드가 가진 에너지를 얼마나 효율적으로 사용하는가 하는 것은 네트워크 전체의 수명을 결정하는 매우 중요한 요소로서 신중히 고려되어야 한다.[4] 따라서 모바일 애드혹 환경에서의 효율적인 에너지 관리와 패킷 전송 시의 응답성, 네트워크의 안정성 등 여러 측면에서의 성능을 향상시키기 위해 다양한 라우팅 기법들이 제안되었다.

MANETs 라우팅 프로토콜은 그 디자인에 따라 크게 몇 가지 형태로 나눌 수 있다. 첫째 선처리(proactive)형과 후처리(reactive)형이다. 노드가 특정 패킷의 전송을 요청하는 시점을 기준으로, 선처리형은 미리 모든 노드로부터 모든 다른 노드로의 경로 정보를 구축하고 전송 요청이 발생하는 즉시 패킷을 전송하는 반면 후처리형은 요청을 받았을 때 비로소 목적지 노드로의 경로를 탐색하여 전송을 하게 된다. 선처리형은 설정된 경로에 변화가 생길 경우, 즉 경로상의 노드의 연결이 끊기는 등의 상황이 발생하면 경로를 다시 설정할 때까지 잘못된 정보를 갖고 있게 된다. 때문에 주기적으로 다시 탐색을 할 필요가 있는데 이로 인해 제어 패킷의 양이 많아지게 된다. 후처리형 라우팅 프로토콜의 관점은 사용될 것인

지 불확실한 경로의 정보까지 수집하는 것은 에너지의 낭비라는 것이다. 전송 요청이 왔을 때 이미 알고 있는 경로라면 다시 보내고, 아니라면 그때부터 경로를 탐색한다. 자연히 즉시 전송하는 선처리방식에 비해 지연 시간이 발생하는 단점이 있으나, 전체적인 효율은 후처리형이 더 높다고 알려져 있다.

다음으로 단순(flat) 구조와 계층적인(hierarcical)구조로 나눌 수 있다. 계층 구조는 네트워크 노드들을 둘 혹은 그 이상의 레벨로 나누고 계층별로 통신하게 하는 방법이다. 네트워크 전체를 대상으로 라우팅하는 단순 구조에 비해 노드의 수가 많아지더라도 비교적 부하가 적게 증가하므로 확장성(scalability)에 있어서 장점을 가진다. 하지만 구조가 복잡해짐에 따라 제어 패킷의 양이 많아진다는 것이 단점이다. 계층 구조의 형태는 트리, zone 등 여러 가지가 있는데 대부분의 프로토콜들은 클러스터를 기반으로 한다.

클러스터 기반 라우팅 프로토콜에서 가장 중요한 작업은 클러스터 헤드의 선출이다. 클러스터 헤드가 클러스터 내의 데이터를 수집하는 역할과 이 데이터를 외부와 주고받는 역할을 함으로써 알고리즘을 단순화하고 네트워크의 효율을 높일 수 있다. 하지만 클러스터의 정보가 헤드로 집중되는 것은 장점이면서 동시에 단점이 된다. 헤드는 에너지를 다른 노드보다 더 빠르게 소모하기 때문에 그 역할이 적절히 분배될 수 있도록 해야 한다. 이로 인해 헤드를 선출하고 선출된 헤드를 중심으로 클러스터 구조를 다시 구축해야하는 부담이 따른다. 또한 패킷이 집중되는 헤드는 네트워크의 병목이 되며, 헤드 노드에 결함이 발생했을 경우 이는 네트워크의 신뢰성을 떨어뜨리는 원인이 된다. 이 때문에 헤드의 결함을 방지 혹은 대처하기 위한 연구들도 진행되고 있다.

이러한 문제점에 대하여 C. Luis, P. David, C.Marilia는 클러스터 헤드 없이 동작하는 클러스터링 기법으로 Novel Stable and Low-maintenance Clustering Scheme(NSLOC)[5]을 제안하였다. 이 기법은 개별 노드의 상태 변화(state transition)에 기반하고 있다. 각 노드는 자신의 이웃 노드와 인접한 클러스터의 상태를 감지하여 자신의 상태를 변화시키고 클러스터 정보를 갱신하여 이를 주변 노드들에게로 전파한다. 이 기법은 헤드의 선출과 클러스터 재구축 과정 없이 안정적으로 클러스터 구조를 만들 수 있지만 클러스터링 알고리즘만을 다루고 있기 때문에 다른 라우팅 프로토콜들과 비교하여 성능의 개선을 이루었다고 속단하기에는 다소 부족하다고 할 수 있다. 또한 각 노드가 임의로 클러스터를 만들거나 기존 클러스터로 편입되기 때문에 전체 네트워크의 관점에서 볼 때 효율적인 클러스터링이 이루어진다고 보기는 어렵다는 등의 문제점도 존재한다.

본 논문에서는 NSLOC 알고리즘을 개선하고, 이를 기반으로 하여 MANETs 에서 안정적이고 효율적인 통신을 위한 클러스터 기반 분산 라우팅 프로토콜(DCRP)을 제안하였다. 이 프로토콜은 클러스터 내부와 클러스터 외부로 가는 패킷을 구분하며 외부로의 전송은 각 클러스터의 게이트웨이 노드들을 통해 이루어진다. 클러스터 내부와 외부의 경로가 분리되어 있기 때문에 다중 경로를 통해 부하를 분산시키게 된다. 또한 DCRP는 완전히 이벤트-드리븐 방식이다. 클러스터의 생성, 노드의 상태 변화, 제어 메시지의 수신, 이웃 노드와 연결이 끊기는 등의 이벤트를 정의하고 네트워크 내의 모든 작업은 규정된 이벤트의 발생과 이에 따른 반응만으로 이루어진다. 설계와 구현이 단순해지며, 이는 결과적으로 네트워크의 유지비용 감소로도 이어진다.

본 논문에서 제안하는 DCRP는 헤드를 선출하는 작업이 필요 없기 때문에 효율적인 에너지 소모가 가능하며, 클러스터를 재구축하는 데 따른 네트워크의 불안정성과 오버헤드를 없앨 수 있다. 2장에서는 이와 관련된 연구 주제들에 대해 살펴보고, 3장에서 DCRP에 대한 구체적인 설명을 하고 4장에서는 이를 구현하여 시뮬레이션을 통해 성능을 평가해 본다. 5장에서는 본문 내용들에 대한 결론을 짓고 앞으로의 방향에 대해 이야기할 것이다. 6장은 참고한 문헌들을 기재하였다.

2. 관련 연구

MANET의 라우팅 프로토콜들에 대해서는 E.Royer의 "A Review of .."[6] 등 여러 논문에 잘 정리되어 있다. MPR(Multi-Point Relay) 팩터를 도입한 Ooptimized Link State Routing(OLSR)[7]이나 거리 벡터(DV)를 통해 라우팅 테이블을 구성 하는 destination-sequenced distance vector(DSDV)[8] 등은 선처리형 프로토콜이며 각 노드 사이의 경로를 캐시하는 Dynamic Source Routing(DSR)[9]과 DSDV를 개선한 Ad-hoc On-demand Distance Vector(AODV)[10] 등은 후처리형 라우팅 프로토콜이다.

계층형 라우팅 프로토콜 가운데서는 Low-Energy Adaptive Clustering Hierarchy(LEACH)[11] 프로토콜 대표적이다. LEACH 프로토콜에 따르면 일정 주기(round)마다 클러스터 헤드를 선출하고 헤드를 중심으로 클러스터를 재구성한다. 한 주기는 클러스터링 단계와 데이터 전송 단계로 나뉘며 클러스터링 단계에 각 노드들은 이웃 노드들과의 통신을 통해 확률적으로 헤드의 역할을 맡아 클러스터 내 노드들의 전송 스케줄링과 데이터 수집(aggregation)을 담당한다. 긴 시간 관점에서 보면 노드들이 번갈아가며 헤드가 될 것을 보장하기 때문에 헤드의 에너지 소모를 균등하게 분배할 수 있지만 클러스터링 단계에는 데이터 전송이 불가능한 점과 제어 메시지의 오버헤드가 있다는 단점이 있다.

그 외에 혼합형(hybrid) 프로토콜들이 존재한다. 이들은 클러스터 내부와 외부를 구분하여 내부에서는 선처리형 프로토콜을, 외부 간의 통신은 후처리형 프로토콜을 사용하는 방식을 통해 효율을 높이려 하였다. 클러스터 기반은 아니지만 혼합형 프로토콜인 Zone Routing Protocol(ZRP)[12]은 각 노드를 중심으로 n홉 내의 노드들이 하나의 구역(zone)으로 구성하는 방식을 제안했다. 따라서 이 구역들은 서로 겹쳐지게 된다. 각 구역 내부의 경로는 link state rouing에 기반한 선처리 방식으로 구성되고 전송 요청이 생기는 즉시 저장된 경로로 패킷을 보낸다. 반면 다른 구역으로의 전송은 후처리 방식으로 진행되는데 경로 요청 단계에서는 송신 노드가 경로 요청을 전파하며 경로 응답 단계에서는 목적지 노드

로의 경로를 알고 있는 노드가 이를 송신 노드로 되돌려 준다. 이 때 경로 요청 메시지는 각 구역의 경계 노드를 따라 전파하는 브로드캐스트에 의해 전달된다. ZRP는 순수한 선처리 혹은 후처리 방식에 네트워크의 트래픽을 줄일 수 있겠지만 하나의 노드가 여러 개의 구역이 겹치게 되므로 이에 대한 처리가 복잡해지는 점과, bordercast로 인해 각 구역 경계의 노드들은 높은 트래픽을 감당해야 하며 최적의 경로를 찾지 못하는 등의 문제가 있다.

ZRP를 기반으로 한 Zone-based Hierarchical Link State(ZHLS)[13]나 Location-Aided Routing(LAR)[14] 등의 프로토콜은 GPS를 사용함으로써 각 노드가 자신의 위치를 정확히 파악할 수 있음을 전제로 하고 있다. 파악된 위치를 바탕으로 손쉽게 구역을 나누고 라우팅 프로토콜을 설계하는 데 주력하고 있다. 하지만 이와 달리 본문에서는 GPS를 사용하지 않는다고 가정하고 있으므로 각 노드가 전체 클러스터 구조 내에서 자신의 위치를 파악하는 것은 굉장히 중요하게 다뤄야 할 이슈이다.

본 논문이 기반을 두고 있는 NSLOC에서는 각 노드가 클러스터 구조와 관련하여 따라 네 가지 상태를 가진다. 또한 노드들은 주기적으로 이웃 노드와의 연결 상태를 파악하고, 이에 따라 자신의 상태를 전이시키고 필요하다면 다른 노드들에게로 메시지를 보낸다. 이 메시지를 받은 노드들은 메시지의 내용에 따라 상태를 변경하거나 메시지를 전파하는 등 클러스터 구조를 관리하는 작업을 수행한다. 이 작업에는 클러스터에 생성하거나 가입하고 클러스터의 크기를 조절하는 과정이 포함된다.

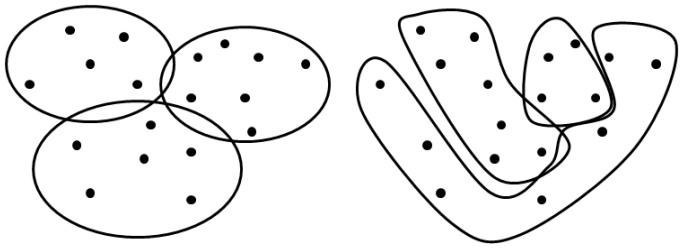

그림 1 클러스터 형태의 비교

NSLOC는 중앙 집중식이 아닌 분산된 방식으로 클러스터를 구성함으로써 기존에 클러스터 구성에 요구되던 오버헤드를 감소시킬 수 있다. 하지만 각 노드가 임의로 클러스터에 가입하며 메트릭이 노드 숫자로만 정의되어 있기 때문에 전체 네트워크 레벨에서 볼 때 효율적인 클러스터링이 이루어지기는 어렵다. 만약 일반적으로 생각하는 형태와는 다른 기형적인 클러스터가 만들어지게 되면(그림 1) 클러스터 내부에 있는 노드 간의 통신보다 다른 클러스터를 경유하는 경로가 더 효율적일 수 있다. 즉 계층 구조의 이점을 얻기 어려운 상태가 된다. 또한 노드의 상태 전이에 따른 동작이 명확하지 않은 부분이 있다. 예를 들어 노드의 이동성이나 고장으로 인해 하나의 클러스터 내에서 특정 두 노드를 잇는 경로가 없어지는 경우, 클러스터 내부의 연결이 완전하지 않으므로 두 개의 클러스터로 나누어지는 것이 옳지만 이와 같은 경우에 대해서는 언급되지 않았다. 본 논문에서는 이러한 문제점을 해결하고, 완전히 분산된 클러스터링을 통해 네트워크의 안정성과 노드 간의 효율적인 라우팅을 가능케 하는 프로토콜을 제안한다.

3. 클러스터 기반 분산 라우팅 프로토콜

MANETs 에서 안정적이고 효율적인 통신을 위한 클러스터 기반 분산 라우팅 프로토콜(DCRP)은 NSLOC 기법을 기반으로 안정적인 클러스터 구조를 형성하며 이를 바탕으로 라우팅이 이루어진다. 클러스터를 구성하는 과정을 전체 네트워크의 관점에서 보이면 다음과 같다.

초기 상태에서는 모든 노드들이 클러스터에 속하지 않은 상태이다. 각각의 노드들은 자율적으로 동작하여 인접한 클러스터가 있다면 가입하고 그렇지 않으면 생성한다. 그리고 생성된 지 얼마 지나지 않아 아직 그리 크지 않은 클러스터들이 합병을 하면서 크기를 키워 간다. 클러스터들이 충분히 커지게 되면 적절한 크기를 유지하기 위해 일부의 클러스터들은 각각 두 개의 클러스터로 분할하게 되며 이러한 과정을 거쳐 클러스터 구조가 안정된 상태에 이르게 된다. 이후로는 주로 이동성이 높은 노드들만이 위치에 따라 클러스터를 변경하고 전체적인 구조는 크게 변하지 않으므로 제어 패킷을 주고받는 오버헤드가 크게 감소한다.

3.1 메시지

노드들은 두 종류의 메시지를 교환하면서 클러스터 구조와 경로 정보를 관리한다. *Ping* 메시지와 *Hello* 메시지가 그것이다. 각 노드는 미리 정해진 시간 간격마다 주기적으로 *Ping* 메시지를 보낸다. 이 메시지는 단지 이웃한 노드와 클러스터의 연결 상태만을 감시한다. *Ping* 메시지를 통해 범위 안에 있던 노드/클러스터의 연결이 끊어지거나, 새로운 노드/클러스터가 들어오는 것을 감지한 노드는 자신의 상태를 변화시키거나 라우팅 테이블을 갱신하고, *Hello* 메시지를 통해 이를 전파한다. *Hello* 메시지는 이벤트가 발생했음을 다른 노드들에게 알리는 메시지이다. *Hello* 메시지 안에는 발생한 이벤트의 종류와 그에 대한 정보가 포함되어 있다. 메시지를 수신하는 노드는 이를 판단하여 처리하고, 필요하다면 메시지를 계속 전파한다.

3.2 노드 상태(node state)

네트워크 상에 존재하는 노드들은 네 가지 중 하나의 상태를 가진다. 노드는 자신이 속한 클러스터를 포함한 연결된 클러스터 수의 카운터인 C_{count}를 가지며, 이 카운터의 값에 의해 상태가 결정된다(그림 2)

Initial : C_{count} = NULL

이 상태는 초기 상태이다. 단지 카운터를 초기화하면서 *Unclustered* 상태로 전이하는 역할만을 수행한다.

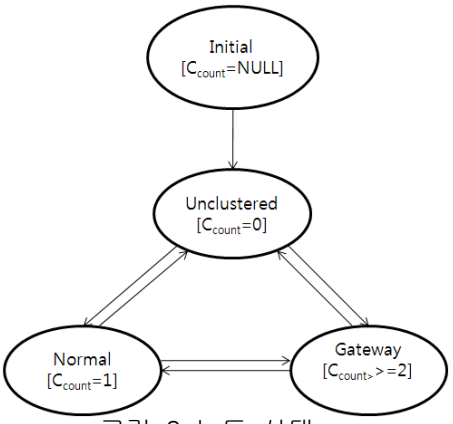

그림 2 노드 상태

Unclustered : C_{count} = 0

클러스터에 속하지 않은 상태이다. 초기화된 직후, 혹은 기존에 속한 클러스터와의 연결이 끊겼을 때 이 상태가 된다. 가입 가능한 클러스터가 있는지를 판단하여, 있다면 가입하고 없을 경우 자신만으로 이루어진 새로운 클러스터를 생성한다. *Normal* 혹은 *Gateway* 상태로 전이한다.

Normal : C_{count} = 1

단 하나의 클러스터와 연결된 상태이다. 즉 클러스터에 속해 있지만 다른 클러스터들과는 직접 연결되지 않는다. 예외적으로 노드의 이동으로 인해 C_{count}값이 1이지만 자신이 속하지 않은 클러스터와 연결되어 있을 경우 *Unclustered*상태가 된다. 연결이 끊어지면 Unclustered 상태로, 다른 클러스터와의 연결이 생기면 Gateway 상태로 전이한다.

Gateway : C_{count} >= 2

다수의 클러스터와 연결된 상태이다. 특정 클러스터에 속해 있으면서 다른 클러스터와도 연결되어 있다. 클러스터 내부와 외부 사이의 통신을 관리한다. 클러스터들과의 연결이 끊어질 경우 유지되고 있는 연결의 수에 따라 Normal 혹은 Unclustered 상태로 전이한다.

3.3 라우팅

DCRP에서 라우팅 테이블은 두 종류가 있다. 거리 벡터(Distance Vector)를 사용하는 노드 레벨 라우팅 테이블(Node level Routing Table, NRT)과 클러스터 레벨 라우팅 테이블(Cluster level Routing Table, CRT)이다. NRT에는 클러스터 내의 노드들에 대해 목적지까지 설정된 경로의 다음 노드가 저장된다. CRT에는 클러스터 별로 해당 클러스터로 향하는 경로의 gateway 노드들을 저장한다. 외부로의 통신은 gateway 노드를 통해서만 이루어지며 각 클러스터 내에서의 전송은 해당 클러스터의 노드들에 의해서 랜덤하게 결정되기 때문에 전체 네트워크의 부하를 분산시킬 수 있다. 그림 3의 예에서 노드 a의 NRT와 CRT를 표1과 표2에 나타내었다.

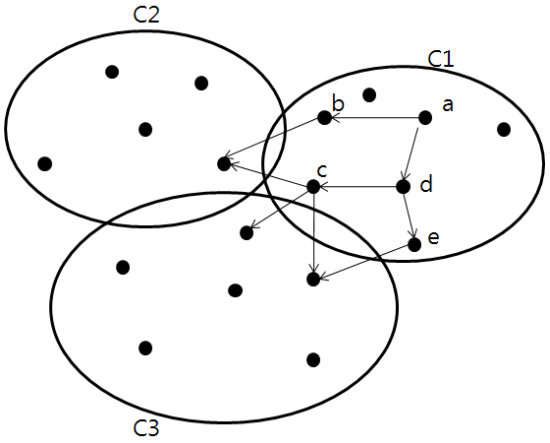

그림 3 클러스터와 라우팅의 예

전송 할 데이터가 있을 때 목표가 클러스터 내의 노드인 경우 NRT 상의 경로를 사용하여 전송한다. 다른 클러스터의 노드라면 CRT를 참조하여 각 클러스터에 질의를 보낸다. 메시지를 받은 다른 클러스터의 gateway들은 목표 노드가 자신의 NRT에 있다면 송신 노드로 응답을 보내고, 송신 노드는 응답이 온 클러스터를 향해 패킷을 전송한다.

표 1 노드 a의 NRT

목적지	다음 노드	hop 수
b	b	1
c	d	2
d	d	1
e	d	2
...	...	...

표 2 노드 a의 CRT

목적지	Gateway
C1	x
C2	b, c
C3	c, e

3.4 연결 상태 감시

클러스터 구조와 라우팅 경로의 관리 작업은 *Ping* 메시지를 통해 노드/클러스터의 연결 상태의 변화를 감지(detection)하는 경우와 다른 노드로부터 *Hello* 메시지를 수신하는 경우에 발생한다. DCRP의 모든 동작은 이를 통해서 이루어지며 노드는 발생한 이벤트의 종류에 따라 정해진 작업을 수행한다.

노드가 이웃 노드와의 연결이 끊어졌음을 감지하는 경우는 두 가지다. 자신의 움직임으로 인한 경우, 클러스터와의 연결이 끊어졌다면 *Unclustered* 상태로 전이하여 다른 클러스터에 가입할 수 있도록 한다. 이웃한 노드의 움직임 혹은 고장으로 연결이 끊어진 경우 주변 노드들에게 *hello* 메시지를 보내 클러스터 내의 경로를 재설정한다. 만약 클러스터 내부의 경로가 완전하지 않다면, 즉 내부의 노드들만을 경유해 서로 통신할 수 없는 노드가 존재한다면 클러스터를 분할하게 된다.

노드가 *Unclustred* 상태에서 *Normal* 상태로 전이하는 경우 *hello* 메시지로 클러스터 내 노드들에게 이를 알리고 NRT와 CRT를 갱신한다. 이 때 거리 벡터를 적용하여 최단 경로 알고리즘을 사용한다. *Unclustred* 상태에서 *Gateway* 상태로 전이하는 경우나 *Gateway* 상태에서 연결된 클러스터가 변하는 경우에는 CRT를 갱신하고 클러스터 내의 노드들에게 이를 알린다.

DCRP는 NSLOC와는 다르게 노드의 수가 아닌 클러스터 내 경로의 최대 홉 수를 메트릭으로 사용한다. 거리 벡터를 사용하는 NRT를 통해 이를 알 수 있다. gateway 노드는 만약 인접한 두 클러스터의 최대 홉 수의 합이 일정치 이하라면 두 개의 클러스터를 병합한다. gwateway 노드의 *hello* 메시지를 받은 두 클러스터 내의 노드들은 이에 따라 자신의 클러스터 ID 값을 변경한다. 반대로, 클러스터 내의 최대 홉 수가 일정치를 넘어서게 되면 클러스터를 분할하여 두 개의 클러스터로 만든다. 이 노드는 Time-To-Live(TTL)값을 설정한 *hello* 메시지를 보내고, 메시지를 받은 노드들은 새로운 클러스터 아이디를 새롭게 바꾼 후 TTL값이 0이 될 때까지 메시지를 재전송한다.

3.5 최적 클러스터 선택 알고리즘

NSLOC 기법의 경우 클러스터를 구성하고 유지하는데 드는 비용을 줄이기 위한 방법을 연구하였다. 하지만 라우팅을 고려하게 되면 이는 문제가 다르다. 안정된 상태가 된 클러스터는 변화가 크지 않으므로, 기형적인 클러스터 구조를 만들게 되는 경우 심각한 성능의 저하를 초래할 수 있다. 한번 기형적인 클러스터 구조를 형성한 후에는 지속적으로 비효율적인 경로를 통해 통신을 하게 된다. 따라서 Unclustered 노드가 클러스터에 가입하려 할 때, 이 노드는 주변의 클러스터 가운데 최대 홉 수가 가장 작은 클러스터를 선택하도록 한다. 이를 통해 DCRP의 클러스터들은 가능한 한 원(circle)형을 유지할 수 있다.

4. 실험 및 평가

제안 기법의 성능 분석을 위해 시뮬레이션을 사용하였다. 시뮬레이터는 NovaSim [15] 을 사용하였다.

클러스터 헤드를 사용하는 클러스터 기반 라우팅 프로토콜인 LEACH [11]와 분산 방식의 제안 기법을 에너지 효율성 면에서 비교하였다. 특히 기존 방식의 LEACH에 이동성을 고려하여 변형한 LEACH'와 비교함으로써 이동성에 따른 성능 분석을 하였다.

토폴로지의 크기는 1kmX1km로 하였고, 노드의 수를 100~300개까지 50개씩 증가시키며 시뮬레이션하였다.

그림 4는 노드의 수가 증가함에 따라 발생하는 추가적인 전송 오버헤드를 나타낸다. 노드의 속도는 5 m/s로 고정하였다. 노드의 수가 증가함에 따라 LEACH'에 비해 DCRP의 오버헤드의 증가량이 적어 확장성이 있음을 확인할 수 있다. 이는 주기적으로 클러스터 헤드의 재선출 및 재구성 비용보다 독립적으로 메시지를 주고받는 분산된 제안 기법이 비용이 적다고 볼 수 있다.

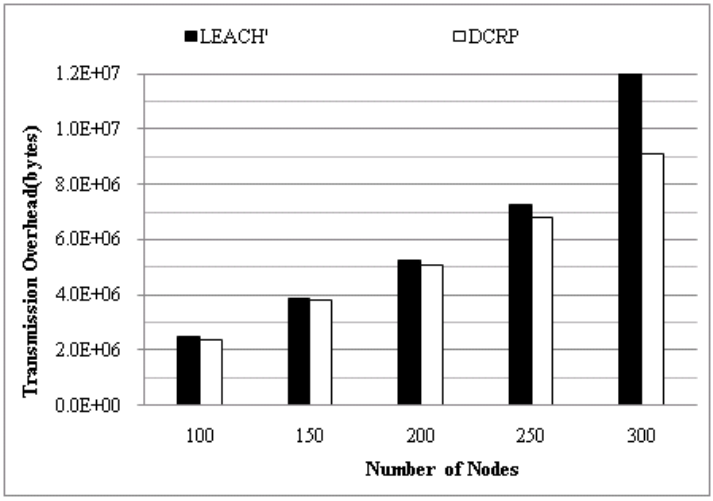

그림 4 노드수에 따른 전송 오버헤드

LEACH'의 경우 노드의 움직임이 빨라질 경우 헤드의 재선출이 잦아지게 되어 Advertisement 메시지와 Join Request 메시지의 양이 잦아 지며, 노드의 수가 많아지기 때문이다.

추가적으로 MANET에서 분산된 방식으로 클러스터를 안정적으로 구성함을 볼 수 있게 하기 위해 속도에 따라 클러스터를 구성하는 노드의 수를 살펴보았다. 1000개의 노드 중에서 각 정해진 시간 주기(60s)로 클러스터 상태의 노드의 수를 그래프로 표현하였다. 노드의 움직임이 빨라질수록 대체적으로 클러스터 상태의 노드가 줄어든다. 하지만, 안정 상태(Steady State)에서는 클러스터 상태의 노드의 수가 안정적으로 존재하는 것을 확인할 수 있다.

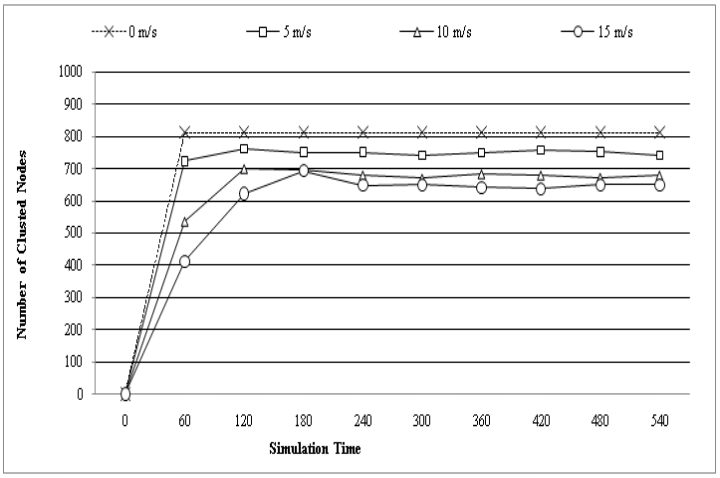

그림 5 제안 기법의 클러스터 구조 안정성

5. 결론

본 논문에서는 클러스터 헤드 없이 클러스터를 구성하는 분산 방식 클러스터 기반의 라우팅 프로토콜(DCRP)을 제안하였다. 제안 기법은 클러스터 헤드의 재선출 및 클러스터의 재구성 비용을 줄이기 위해 클러스터 헤드의 선출을 통한 중앙 집중식의 접근 방식이 아닌 독립적으로 자신의 상태를 변화시킴으로써 클러스터를 구성하는

방법을 사용한다. 시뮬레이션을 통해 제안 기법이 안정적인 네트워크 구조와 효율적임을 확인하였다.

향후 움직임 모델을 고려하여 더욱 효율적이고 안정적인 분산 클러스터 기반의 라우팅 프로토콜을 연구할 것이다.

6. 참고 문헌

[1] Römer Kay, Friedemann Mattern, "The Design Space of Wireless Sensor Networks". IEEE Wireless Communications, 11, 6, pp. 54-61, 2004.

[2] Thomas Haenselmann, "Sensornetworks", GFDL Wireless Sensor Network, 2006.

[3] Hadim Salem, Nader Mohamed, "Middleware Challenges and Approaches for Wireless Sensor Networks", IEEE Distributed Systems, 2006.

[4] Rahul C. Shah and Jan M. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks", Proc. IEEE Wireless Communication and Network Conference (WCNC), vol. 1, pp. 350-55, 2002.

[5] Luis Conceicao, David Palma, Marilia Curado, "A Novel Stable and Low-maintenance Clustering Scheme", ACM Symposium On Applied Computing, 2010.

[6] E.Royer, C.-K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", IEEE Personal Communications, pp. 46-55, 1999

[7] T. Clausen and P. Jaquet. "RFC 3626 – Optimized Link State Routing Protocol (OLSR)". http://www.faqs.org/rfcs/rfc3626. html, 2003

[8] Perkins, C. E., Bhagwat, P. "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", Computer Communications, pp. 234-244, 1994.

[9] Johnson, D. B.,Maltz, D. A., "Dynamic Source Routing in Ad-Hoc Wireless Networking", Mobile Computing, T. Imielinski and H. Korth, Eds. Norwell, MA: Kluwer, pp. 153-181, 1996.

[10] Perkins, C. E., Royer, E. M., "Ad-hoc On-Demand Distance Vector Routing", Proc. 2nd IEEE Workshop on Mobile Computer Systems and Applications, pp. 90-100, 1999.

[11] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan, "Energy-Efficient Communication Protocol forWireless Microsensor Networks", Proc. of the 33rd Hawaii International Conference on System Sciences, 2000.

[12] Z. J. Haas, "The zone routing protocol (ZRP) for ad hoc noetworks", Internet Draft, 1997

[13] Mario Ja-NG, I-Tai Lu, "A Peer-to-Peer Zone-Based Two-Level Link State Routing for Mobile Ad Hoc Networks", IEEE journal on selected areas in communications, vol. 17, no. 8, pp. 1415-1425, 1999

[14] Young-Bae Ko and Nitin H. Vaidya, "Location-Aided Routing (LAR) in mobile ad hoc networks", Wireless Networks, 6, pp. 307-321, 2000.

[15] Jinman Jung and Yookun Cho , "Virtual Protocol Stack Interface for Multiple Sensor Network Simulators", ACM SAC 2010.