개미 시스템을 기반으로 한 Ad hoc 네트워크 멀티캐스팅

Ad hoc Network Multicasting Algorithm Based on An Ant System

이 세 영*, 김 중 항, 장 형 수
(Se-young Lee, Joong Hang Kim, and Hyeong Soo Chang)

Abstract : This paper proposes a novel multicasting algorithm, called ANMAS (Ad hoc Network Multicasting with Ant System), for Mobile Ad hoc Network (MANET). The algorithm utilizes the indirect communication method of the ants via "pheromone" to effectively obtain dynamical topology change information, generating safer multicasting paths, and adapts the well-known CBT (Core Based Tree) multicasting algorithm into the ANMAS framework with proper modificiations to make "tolerable" multicasting group in the MANET environment. We show the efficiency and the effectiveness of ANMAS via simulation studies.

Keywords : Ad hoc network, multicast, ant system, swarm intelligence, ODMRP

I. 서론

본 논문에서는 유선 네트워크 상의 멀티캐스팅 알고리즘인 CBT(Core Based Tree)[1]를 Mobile Ad hoc Network(MANET)[2]에 맞게 적절히 변형하고, 여기에 "개미 집단 최적화(Ant Colony Optimization)" 알고리즘[3](이하 개미 알고리즘)의 특성을 융합한 ANMAS(Ad hoc Network Multicasting with Ant System) 알고리즘을 제안하고자 한다.

MANET는 노드의 이동에 의한 위상 변화가 잦은 반면, 전파 특성에 의해 한번 전송한 데이터를 주변의 여러 노드가 동시에 받을 수 있다는 장점(broadcast 특성)이 있다. 기존 MANET상의 멀티캐스팅 알고리즘 중에서는 broadcast 특성을 이용하는 mesh 방식(대표적으로 On-demand Multicast Routing Protocol(ODMRP)[4])이 unicast를 기본으로 하는 tree 방식(대표적으로 Ad hoc Multicast Routing protocol utilizing Increasing id-numberS(AMRIS)[5])에 비해 더 효율적인 것으로 알려져 있다[6]. 특히 ODMRP의 경우 주기적인 경로 재구성을 통해 네트워크의 위상변화를 반영함으로서 여타 알고리즘에 비해 나은 데이터 패킷 전송율을 보인다. 하지만 경로 붕괴를 감지하여 재구성할 수 있는 방법이 없다는 점과 위상 정보를 수집, 종합하여 보다 안정적인 경로를 형성할 수 있는 방법이 없다는 점 등이 단점이라 할 수 있다[4].

ANMAS는 mesh 방식의 장점을 수용하고, 여기에 CBT 알고리즘을 적절히 변형, 적용함으로서 경로 붕괴를 감지, 재구성하여 보다 견고한 그룹을 구성할 수 있으며, 개미 알고리즘의 특성을 융합하여 얻어지는 위상 정보를 통해 안정적인 멀티캐스팅 경로를 생성할 수 있다. 특히 ANMAS는 송신자(source) 중심의 그룹을 구성하면서도 개미 알고리즘의 간접적 정보전달 특성을 이용하여 적은 패킷 부하만으로 다수의 송신자가 그룹을 구성할 수 있다.

이미 CBT 알고리즘에 기초하여 만들어진 CAMP (Core-Assisted Mesh Protocol)[7], CBT에 개미집단 알고리즘을 적용한 MANSI (Multicast for Ad hoc Network with Swarm Intelligence)[8] 알고리즘 등이 나와 있으나, 이들 모두 기존 ODMRP의 성능에 미치지 못한다[6,8]. ANMAS는 앞서 열거한 특성을 통해 기존의 ODMRP에 비해 효율적이고 견고한 멀티캐스팅 그룹을 구성할 수 있으며, 실험 결과를 통해 이를 확인할 수 있다.

본 논문의 구성은 다음과 같다. II장에서는 개미 알고리즘과 그 특성에 대해 간략히 소개하고, MANET 상의 멀티캐스팅에 유용함을 서술한다. III장에서는 ANMAS 알고리즘을 CBT 알고리즘 및 개미 알고리즘과 연계시켜 설명한다. IV장에서는 실험 결과를, 마지막으로 V장에서는 본 논문의 결론을 내린다.

II. 개미 집단 최적화 알고리즘(Ant Colony Optimization)

자연계의 집단생활을 하는 생물들 중에서는 각 개체의 지능으로는 해결하지 못하는 고차원적인(higher order) 문제를 그들만의 독특한 방식으로 해결해 나가는 것을 관찰할 수 있는데 이러한 특성을 이용하여 만들어진 알고리즘을 군지능 알고리즘(Swarm Intelligence)[9]이라 한다. 군지능 알고리즘의 하나인 개미(집단 최적화) 알고리즘은 자연계의 개미들이 둥지와 먹이 간에 최단거리를 찾아 이동하는 습성을 관찰하여 만들어진 알고리즘이다. 개미 알고리즘에서는 주어진 문제의 최적해를 포함한 가능해들의 집합을 그래프로 나타내고, 이 위에 인공개미를 확률적으로 이동시키면서 그래프 상의 경로로 이루어진 후보해를 생성한다. 각 개미는 생성한 후보해의 질을 평가하여 해의 우수성에 맞는 양의 페로몬을 경로에 묻히는데, 이는 우수한 경로를 탐색할 확률을 높이는 역할을 한다. 개미의 이동 경로는 페로몬과 개별 경로가 가지는 가중치의 조합을 통해 확률적으로 결정되며, 후보해를 찾고 우수한 해에 대해 더 많은 페로몬을 더하는 과정을 반복함으로서 개미들이 목적한 최적해에 수렴하도록 한다.

본 논문에서는 개미 알고리즘의 페로몬을 통한 간접적인 정보 전달 방식에 주목한다. MANET 노드의 이동성으로 인

* 책임저자(Corresponding Author)
논문접수 : 2004. 9. 20., 채택확정 : 2004. 10. 26.
이세영, 김중항, 장형수 : 서강대학교 컴퓨터공학과
(philolight@sogang.ac.kr/peropero@sogang.ac.kr/hschang@sogang.ac.kr)
※ 이 논문은 한국학술진흥재단의 지원에 의하여 연구되었습니다(KRF-2004-003-D00294).

한 위상 변화는 라우팅 및 멀티캐스팅 경로의 변화를 유발하기 때문에 효율적인 데이터 전송을 위해서는 이를 수시로 파악해야만 한다. 그러나 이러한 정보를 각 노드에 전달하는 것은 적은 대역폭이나 전원 문제로 인해 어려움이 많다. 개미 알고리즘에서는 개미가 찾은 해들을 평가하여 페로몬화 한 후 이를 그래프의 경로 위에 묻힘으로서 '가상의 해 평가 그래프'를 구성한다. 그리고 개미들은 이 그래프 위의 노드가 가진 지역정보만으로 이동을 계속하면서 유용한 해들을 생성해낸다. MANET에서 멀티캐스팅에 필요한 위상정보를 개미 알고리즘의 가상 그래프처럼 구성하고 이를 사용할 수 있다면 최소한의 패킷 전송만으로도 효율적인 멀티캐스팅 그룹을 구성할 수 있을 것이다. 따라서 ANMAS는 멀티캐스팅에 유용한 위상정보를 각 이웃노드(다른 노드를 거치지 않고 직접 통신 가능한 노드)에게 얻은 후 이를 평가하여 다시 이웃노드에게 전달하는 방식으로 '가상의 경로 정보 그래프'를 구성하여 각 노드가 가진 지역정보만으로도 유용한 멀티캐스팅 경로를 형성할 수 있도록 한다. 이에 대한 구체적인 내용은 다음 장에서 자세히 설명하도록 한다.

III. ANMAS
(Ad hoc Network Multicasting with Ant System)

1. ANMAS의 그룹 구성 방법

ANMAS는 mesh 방식에 따라 "forwarding set"이라 부르는 멀티캐스팅 그룹을 구성한다. 그룹 구성에 있어서 기본적으로 CBT 알고리즘[1]을 따르며, 다음과 같은 세 단계로 이루어진다(그림 1 참조).

1.1 Core Announce 단계

CBT 알고리즘에서는 보통 최초의 참여자 노드(멀티캐스팅 데이터를 송신, 혹은 수신하려는 노드)가 최초의 그룹 구성자이자 그룹 회합점 역할을 하는 Core가 되지만 ANMAS에서는 송신자(source)가 Core 역할을 한다. 최초의 송신자는 네트워크 내에 이미 멀티캐스팅 그룹이 구성되어 있는지를 알아보고, 그렇지 않다면 Core Announce 메시지를 flooding하여 자신이 Core임을 알린다(그림 1.(1)). Core는 다른 노드들의 그룹 참여 요구를 수락하고, 정기적으로 Core Announce 메시지를 flooding 함으로서 그룹이 지속되고 있음을 알리는 역할을 한다.

1.2 Join Request 단계

어떤 노드가 새로 그룹에 참여하고 싶다면 이 노드는 자신의 이웃노드 중 하나를 선택하여 Core에게 그룹 참여를 요구하는 Join Request 메시지를 전달한다(그림 1.(2)). 이 메시지를 받은 노드가 Core가 아니면 다시 신의 이웃노드 중 하나를 선택하여 이 메시지를 전달하고, 자신이 Core이면 그룹 참여를 수락하는 Join Acknowledge 메시지를 전달한다.

1.3 Join Acknowledge 단계

Core가 Join Request 메시지를 받았다면 그룹의 참여를 수락하는 Join Acknowledge 메시지를 보낸다(그림 1.(3)). 이 메시지는 Join Request 메시지가 전달된 경로의 반대 방향으로 이동하고, 이 메시지를 받은 노드가 참여자 노드가 아니라면 참여자간 연결을 담당하는 forwarding 노드로 그룹에 포함된다. 최종적으로 참여자 노드에게 이 메시지가 전달되면

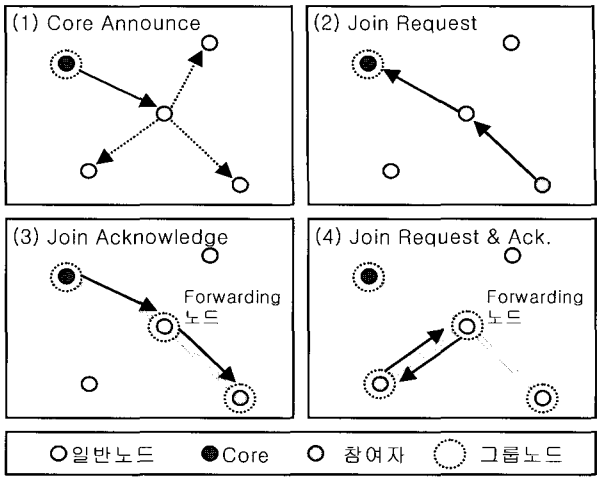


그림 1. ANMAS의 그룹 구성 방식(CBT와 동일).
Fig. 1. Group construction method of ANMAS.

참여자 노드 역시 그룹의 일원이 된다. Core를 포함한 멀티캐스팅 그룹 내의 모든 노드는 다른 노드들에 의한 참여 요구를 수락할 수 있다(그림 1.(4)).

ANMAS는 이미 그룹을 구성하고 있는 어떤 노드라도 Join Request를 수락하여 빠르게 그룹을 구성할 수 있게 하는 CBT 알고리즘의 특성을 그대로 가진다. Join Request 메시지의 경로는 멀티캐스팅 그룹의 경로와 직결되기 때문에 어떤 경로를 선택하여 메시지를 보낼 것인가가 멀티캐스팅 효율을 좌우하는데, CBT 알고리즘에서 모든 노드는 Core를 향해 Join Request를 보내므로 각 노드가 Core와의 거리 및 그 경로에 대한 정보를 알 수 있다면 효율적인 그룹을 구성할 수 있다. ANMAS는 이 Core에 대한 정보를 페로몬으로 저장하는데, 2절에서는 이를 이용한 멀티캐스팅 경로 구성 방법을 알아보기로 한다.

2. 개미 알고리즘을 이용한 멀티캐스팅 경로 구성

일반적인 개미 알고리즘은 정적인 그래프 위에 페로몬을 묻히고, 더 나은 해에 대하여 더 많은 페로몬을 묻히는 행위를 반복함으로서 최적해에 수렴하도록 하는 방식을 취한다. 하지만 MANET와 같은 동적인 환경에서 이러한 수렴성은 위상 변화에 대한 능동적인 대처를 방해할 수도 있다. 따라서 ANMAS는 개미 알고리즘의 수렴성보다는 각 개미들이 간접적인 의사소통 수단으로 사용하는 페로몬과 이에 대한 적절한 연산을 통해 위상 변화를 반영할 수 있는 유용한 정보를 생성하는데 초점을 맞춘다.

ANMAS는 각 노드를 개미로 보고, CBT 알고리즘에서 경로 생성의 중추 역할을 하는 Core에 대한 정보를 페로몬화하여 상호 교환한다. 각 개미(노드)는 이웃개미들의 페로몬을 종합하여 자신의 페로몬을 계산한 후 다시 이웃개미에게 알린다. 또한 이웃개미들의 페로몬을 따로 저장하여 우수한 페로몬을 가진 이웃을 선택할 수 있게 한다. 모든 개미가 이와 같은 페로몬 연산과 전달을 반복하고, 이웃 개미들에 대한 정보를 따로 저장함으로서 이웃과 자신의 지역적(local) 정보로만 이루어진 '가상의 경로 정보 그래프'를 구성한다. 각 노드(개미)가 계산하는 페로몬 정보는 크게 두

가지로 구성된다. 하나는 Core와 자신과의 거리를 의미하는 Dc이고, 다른 하나는 자신과 Core 사이의 경로가 얼마나 안정적인가를 표현하는 Pc이다. 안정적인 경로란 경로붕괴의 위험이 적은 경로를 말한다. Dc와 Pc를 통해 Join Request 경로를 선택함으로서 보다 짧고 안정적인 경로를 구성하게 된다. 기존 알고리즘 중에는 GPS(Global Positioning System) 정보를 이용하여 노드의 이동성을 파악하고 멀티캐스팅 경로 붕괴에 대비하는 알고리즘[10]이 나와 있지만 ANMAS는 그러한 정보를 가정하지 않는다. GPS 정보를 이용하지 않는다는 점은 이를 이용할 기반시설이 갖추어지지 않은 지역에서도 효과적인 멀티캐스팅이 가능하다는 장점을 갖게 한다.

2.1 Dc의 용도 및 계산 방법

노드 i의 Dc를 Dc_i라 하고 이는 Core와 노드 i간의 거리(hops)를 나타낸다. Dc_i는 다음과 같이 계산된다.

$$Dc_i = \begin{cases} 0 & \text{if } i = Core \\ min_{j \in N_i}\{Dc_{j \to i} + 1\} & else \end{cases} \quad (1)$$

N_i는 i의 이웃노드 집합이고, $Dc_{j \to i}$는 노드 i가 이웃노드 j로부터 전달받은 Dc이다. 노드 i가 Core일 경우 Dc_i는 항상 0이다. Core의 이웃노드 j는 Core의 Dc가 가장 작은 값(0)이므로 Dc_j를 1로 계산한다. 다른 노드들은 이웃노드들의 Dc로 자신의 Dc를 계산하는데, 노드 k의 이웃노드가 가진 Dc 중에서 가장 작은 수가 n이라면 Dc_k는 n+1이 된다.

어떤 노드가 다른 노드에 비해 작은 Dc값을 가진다면 그 노드를 통해 Join Request를 보낼 경우 더 짧은 경로를 구성할 가능성이 높고, 이는 그룹의 효율성을 높이므로 낮은 Dc를 갖는 노드를 통해 우선적으로 Join Request 메시지를 보내도록 한다.

2.2 Pc의 용도 및 계산 방법

Pc는 각 노드가 자신의 이웃노드들로부터 받은 정보를 통해 Core와 자신의 경로에 대한 안정성을 계산하여 수치로 나타낸 값이다. 안정성은 본 논문에서 새롭게 도입하고자 하는 개념으로서, "평균 통신 지속 시간"을 통해 MANET 노드의 이동성에 의한 경로 붕괴가 일어날 가능성을 평가하는 척도이다.

평균 통신 지속 시간(Ts_i)는 다음과 같이 정의된다.

$$Ts_i = \sum_{\lambda_i \subset N_i} \sum_{j \in \lambda_i} \frac{Tc - Tf_i(j)}{|\lambda_i|} \; (|\lambda_i| \geq 1) \quad (2)$$

λ_i는 N_i의 부분집합이고 $|\lambda_i|$는 이 집합 내 노드의 수이다. Tc는 현재 시간이고 $Tf_i(j)$는 노드 j가 i의 이웃노드로 등록된 시간이다. 따라서 (2)는 λ_i에 속한 노드들이 노드 i와 통신한 시간을 평균한 값, 즉 평균 통신 지속 시간(초)이다. 이 식을 통해 알고자 하는 것은 각 노드들의 이동성이다. 노드의 이동에 의한 위상변화는 각 노드들 간의 통신에 가장 큰 영향을 미치는 요소지만, 불행히도 GPS 등의 외부적인 정보 없이는 자신이나 다른 노드의 위치를 알 수 없으므로 각 노드의 이동을 직접 판단할 수는 없다. 따라서 본 논문에서는 Ts_i를 이용하여 간접적인 방법으로 이동성에 대한 정보를 얻고, 이를 안정성에 반영하고자 한다.

먼저 모든 노드가 동일한 통신 가능 영역(propagation range)를 가지고 있고, 어떤 두 노드가 서로 이 영역 내에 있을 경우에만 통신이 가능하다면 다음과 같은 예측이 가능하다.

예측 1 : 같은 지점에서 출발한 두 노드가 동일한 상대속도(두 노드의 속도 벡터 차)로 움직일 때 이들의 상대적인 거리는 시간에 비례하여 커진다. 통신 가능 영역 내에 있는 두 노드가 일정한 상대속도를 유지한다면 시간이 지날수록 서로 통신 가능 영역을 벗어날 가능성이 커질 것이다. 따라서 두 노드 간의 통신 지속 시간이 커질수록 통신이 끊어질 가능성은 높아질 것이다.

예측 2 : i라는 노드가 [0, V_{max}]의 속도로 움직이고 다른 노드들 역시 [0, V_{max}]의 속도로 움직일 때 i와 다른 노드들 간의 상대속도는 [0, $2V_{max}$]의 분포를 가지게 된다. 반면 i가 움직이지 않을 경우에 상대속도는 [0, V_{max}]의 분포를 가진다. 따라서 i가 움직이지 않는 상태를 지속적으로 유지할 경우 다른 노드와 오래동안 통신을 지속할 가능성이 높고, 따라서 i는 움직이는 노드들에 비해 오래 동안 통신을 지속한 이웃노드들을 더 많이 가지고 있을 것이다. 결국 움직이지 않는 노드일수록 이웃노드들과의 평균 통신 지속 시간이 더 클 것이다. 반면 움직이는 노드는 통신 가능 영역이 지속적으로 변하고, 새로운 통신 가능 영역에 있는 노드와 통신을 시작할 가능성이 높으며, 기존의 통신 가능 영역에서 오랫동안 통신을 지속해왔던 노드와의 연결이 끊어질 가능성이 높을 것이다. 따라서 움직이는 노드는 이웃노드들과의 평균 통신 지속 시간이 짧을 것이다.

예측 1, 2와 (2)를 통해 노드 i의 특정한 이웃노드 j가 얼마나 안정적인가와 노드 i 자신이 얼마나 안정적인가를 알고자 한다. 전자를 이웃노드의 안정성이라 하고 후자를 자기 노드의 안정성이라 한다. 먼저 이웃노드의 안정성은 λ_i를 특정 이웃노드 j만이 속한 집합으로 정의하였을 때 계산되는 Ts_i, 즉 i와 j의 통신 지속 시간에 의해 좌우되고, 이러한 경우에 Ts_i를 Ts_i^j라 한다. 예측 1에 의해 Ts_i^j가 큰 이웃노드 j는 통신 시간이 짧은 이웃노드에 비해 경로 붕괴의 위험이 높다고 예상할 수 있다. 따라서 각 노드는 Ts_i^j가 작은 이웃노드일수록 안정적인 노드로 판단한다.

다음으로 자신의 안정성은 λ_i를 모든 이웃노드 즉, N_i와 동일하게 놓고 계산하여 나온 Ts_i를 중심으로 판단되고, 이 경우의 Ts_i를 Ts_i^A라 한다. 예측 2에 의하여 노드 i가 큰 Ts_i^A를 가지고 있다면 이 노드는 상대적으로 느리게 움직일 가능성이 높고, 따라서 다른 노드들과의 통신을 오래 지속할 가능성이 높으므로 스스로 안정적이라고 판단한다.

예측 1, 2를 바탕으로 한 안정성 계산은 상반성을 가지고 있다. 특히 $|N_i|$가 1일 경우, 즉 i의 이웃노드가 하나일 경우 Ts_i^A와 Ts_i^j는 같은 값을 가지게 되지만 이를 통한 안정성 판단은 반대가 된다. 실제적으로 예측 1, 2는 개연성을 가지기는 하지만 부족한 정보를 바탕으로 한 완벽하지 않은 예측이다. 따라서 서로 모순되는 예측 1과 2는 상호보완적인 형태로 사용되어야 한다. 즉, 개별 쌍의 안정성 평가는 Ts_i^j와 예측 1에 의존하지만 Ts_i^A와 예측 2를 통해 노드 i의 안정성 정보를 추가함으로서 보다 정확한 안정성 평가를 하도록 도울 수 있다. 반대로 노드 i가 자신의 안정성을 계산

할 때 개별 쌍의 안정성에 대한 정보를 이용하는 것이 더 정확한 평가를 가능하게 한다. 이에 대한 구체적인 방법은 수식 (4),(5)를 통해 보여주기로 한다. 노드 i는 Ts_i^A와 Ts_i^j, 그리고 이웃노드들의 안정성을 통해 자신의 안정성을 계산하여 이웃노드들에게 알려주고, i의 이웃노드 j가 스스로의 안정성을 계산하여 알려준 값을 재평가한다.

이웃노드 j가 스스로를 평가하여 i에게 보낸 안정성 값, $Pc_{j\to i}$는 노드 i의 이웃노드 정보에 그대로 저장되지만 노드 i는 이를 Ts_i^j에 따라 감소시킨 값을 따로 계산하여 사용한다. 즉, 예측 1에 의해 두 노드간의 통신시간이 길수록 통신이 끊어질 가능성이 높다는 것을 반영하는 것이다. $Pc_{j\to i}$의 감소는 일반 개미 알고리즘의 페로몬 증발(evaporation)을 응용한다.

$$\tau_{ij}(t+1)=\tau_{ij}(t)(1-\rho)\ \ (0<\rho<1) \tag{3}$$

τ는 페로몬의 양이며, τ_{ij}는 그래프 상의 노드 i에서 j 로 향하는 경로 상에 묻어 있는 페로몬의 양이다. 개미 알고리즘에서는 매 반복(iteration)마다 후보해의 경로에 페로몬을 부여하는 동시에 (3)에 의한 페로몬 증발을 수행하여 우수한 해에 대한 수렴도를 높인다. ρ는 개미 알고리즘에서 말하는 페로몬 증발율(evaporation factor)로서 페로몬이 증발하는 양을 조절한다. 개미 알고리즘은 후보해를 생성하고 페로몬을 부여하는 주기를 반복하는데 t는 이 주기를 반복한 횟수이다. 본 논문에서는 이 t를 실시간으로 확장하여 $Pc_{j\to i}$를 시간에 따라 감소시키는 함수로 변형하여 사용하고, 이를 $Pc_i(j, Ts_i^j)$라 한다. $Pc_i(j, Ts_i^j)$는 (4)와 같이 계산된다.

$$Pc_i(j,Ts_i^j)=Pc_{j\to i}(1-\mu)^{T_{ij}}\ (0<\mu<1)$$
$$(T_{ij}=max\{0,\ Ts_i^j-Ts_i^A\}) \tag{4}$$

(4)에 의해 통신 지속 시간이 큰 노드일수록 $Pc_i(j, Ts_i^j)$의 값은 작아지지만 노드 i의 평균 통신 지속 시간(Ts_i^A)이 크다면 그 감소 속도는 작아지게 된다. 즉, Ts_i^A가 큰 노드 i는 앞으로도 이웃노드와 통신을 지속할 가능성이 크다고 판단하고 $Pc_i(j, Ts_i^j)$의 감소 속도를 늦춘다. 만일 i가 단 하나의 이웃노드 j를 가지고 있을 경우 Ts_i^j와 Ts_i^A는 같아지고, $Pc_i(j, Ts_i^j)$는 $Pc_{j\to i}$를 그대로 따름으로서 이웃노드 j에 대한 안정성 판단을 j에게 유보한다. 이를 통해 안정성 판단의 모순성을 제거하고 보다 더 많은 이웃노드를 가지고 있을 가능성이 높은 j의 판단을 따르게 된다. μ는 페로몬 증발율(ρ)을 응용한 안정성 감소율로서 이 값이 크면 클수록 새로운 이웃노드의 안정성을 더 높게 평가하게 된다.

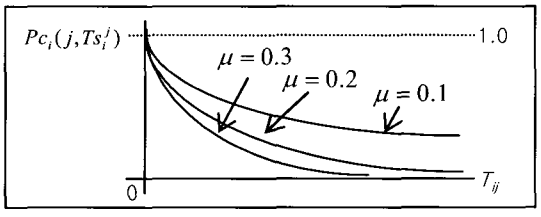


그림 2. $Pc_{j\to i}$ = 1.0일 때 $Pc_i(j, Ts_i^j)$ 그래프.
Fig. 2. Graph of $Pc_i(j, Ts_i^j)$ for $Pc_{j\to i}$ = 1.0.

μ는 heuristic하게 결정되는데, 반복적인 실험에 따르면 [0.01, 0.15] 구간의 값을 사용하는 것이 효율적이다(IV장의 표 1 ~ 3 참조). 이렇게 계산된 $Pc_i(j, Ts_i^j)$는 Join Request 메시지 전달 방향을 결정하거나 노드 i의 안정성(Pc_i)을 계산하는데 사용된다. 그림 2는 $Pc_{j\to i}$가 1.0일 때 T_{ij}에 따라 $Pc_i(j, Ts_i^j)$가 어떻게 변하는가를 보여주는 그래프이다.

Pc_i는 이웃노드들에 대한 안정성$Pc_i(j, Ts_i^j)$과 Dc 즉, Core와의 거리를 반영하여 계산된다. Pc_i는 i와 Core와의 경로의 안정성을 반영하는 것이므로 노드 i가 자신보다 작은 Dc를 가지는 이웃, 즉 자신보다 Core와의 거리가 가까운 이웃노드가 많을수록, 그리고 그러한 이웃노드들이 통신을 지속한 시간이 짧을수록 높은 안정성을 가지도록 해야 한다. 노드 i의 안정성을 반영하는 Pc_i는 다음과 같이 계산한다.

$$Pc_i=\begin{cases} c & (c>0) \quad \text{if } i=Core \\ \displaystyle\sum_{j\in N_i, Dc_{j\to i}<Dc_i} Pc_i(j,Ts_i^j) & else \end{cases} \tag{5}$$

c는 양의 실수 중 하나를 취하여 사용한다. Core는 항상 일정한 Pc값, c를 이웃에 전달하고, 이외의 노드는 이웃노드들이 보내준 값으로 자신의 Pc를 계산한다. (5)에 따라 이웃노드들 중에서 자신의 Dc_i보다 더 작은 Dc를 가지는 노드가 많을수록, 그리고 그 노드들이 자신과 통신을 지속한 시간이 짧을수록 Pc_i는 커진다. 또한 $Pc_i(j, Ts_i^j)$가 내부적으로 Ts_i^A를 사용하므로 (4) 자신의 안정성 Ts_i^A가 클수록 더 높은 Pc_i 값을 가지게 된다. Dc와 Pc는 3절에서 설명할 Pheromone Update 메시지에 의해 정기적으로 이웃에게 보내지고, 이를 통해 간접적인 Core 정보를 얻게 된다.

이렇게 얻어진 Dc와 Pc 정보는 Join Request 메시지를 보낼 이웃노드를 결정하는데 사용된다. 노드 i가 다른 노드에게 Join Request를 보내야 할 때에, 어떤 이웃노드(k)에게 보낼 것인가는 (6)에 의해 결정된다.

$$k=\underset{j\in N_i, j\notin visited}{argmax} Pc_i(j,Ts_i^j)\cdot\alpha^{Dc_{j\to i}-Dc_i+1}$$
$$(0<\alpha<1) \tag{6}$$

*visited*는 다른 노드로부터 받은 Join Request를 보내야 할 때 이미 이 메시지가 방문했던 노드의 집합이고, 이미 방문한 노드에는 Join Request를 다시 보내지 않는다. α는 단거리 경로에 대한 수렴도로서 이 값이 작을수록 자신의 Dc보다 작은 Dc를 가지는 이웃노드가 Join Request를 더 많이 받게 한다. α 역시 실험을 통해 heuristic하게 결정되고, [0.01, 0.15] 구간의 값이 효율적이다(IV장의 표 1, 2, 3 참조). (6)에 의해 Core에 가깝고, 보다 안정적인 노드일수록 Join Request 메시지를 받을 가능성이 높아지게 된다. k를 결정하는 식 역시 일반 개미 알고리즘에서 개미가 후보해를 생성하면서 어떤 이웃노드를 선택하여 이동할 것인가를 결정하는 (7)을 응용한 것이다.

$$probability(i,j)=\frac{[\tau_{ij}(t)]^\delta[\eta_{ij}]^\theta}{\displaystyle\sum_{j,l\in N_i}[\tau_{il}(t)]^\delta[\eta_{il}]^\theta}\ (\delta,\theta\geq 0) \tag{7}$$

일반적인 개미 알고리즘에서는 (7)을 이용하여 각 이웃노드 j에 대하여 이동할 확률을 계산한다. 노드 i에서 j로 가는 경로상의 페로몬(τ_{ij})이 많으면 많을수록, 그리고 그래프상의 가중치(η_{ij})가 높으면 높을수록 개미가 j로 이동할 확률은 높아진다. 이와 약간 다르게 k는 Dc와 Pc에 의한 조합(6)이 가장 큰 노드로 결정된다.

ANMAS는 Core와의 거리를 나타내는 Dc와 경로의 안정성을 나타내는 Pc를 이용하여 Join Request 방향을 결정함으로서 보다 효율적이고 안정적인 그룹을 구성하게 된다. 하지만 아무리 안정적인 그룹을 형성하였다 하더라도 경로의 붕괴를 감지하고 대처할 필요가 있고, 또한 시시각각 변화하는 네트워크의 위상 정보를 통해 정기적으로 경로를 재구성할 필요가 있다. 따라서 3절에서는 경로 붕괴에 대한 대처 및 정기적인 경로 재구성을 통한 그룹 유지에 대해 살펴보고자 한다.

3. 멀티캐스팅 그룹 유지

ANMAS에서는 CBT 알고리즘의 경로 붕괴 감지 방식을 MANET에 맞게 변형하여 적용한다. ANMAS에서는 Hello 메시지를 통해 경로 붕괴를 감지하고 Flush Tree 메시지를 통해 경로를 재구성하도록 한다. 이들 메시지에 대한 보다 상세한 설명은 다음과 같다.

· Hello(Pheromone Update)

각 노드는 Hello 메시지를 정기적으로 이웃노드들에게 보내어 자신이 주위에 있음을 알리며, 이웃노드에게 일정 기간 이 메시지를 받지 못했다면 그 노드는 자신의 주위에서 사라진 것으로 간주한다. ANMAS에서는 Hello 메시지가 페로몬 전달 역할을 수행하므로 Pheromone Update 메시지라 한다. 노드 i가 이 메시지를 통해 전달하는 Dc_i와 Pc_i는 이웃노드들에게 노드 i가 얼마나 Core와 가깝고 안정적인지를 알 수 있도록 해준다. 또한 노드 i는 다시 이웃노드들로부터 얻은 Dc와 Pc를 종합하여 자신의 Dc와 Pc를 계산(식 (5)), 다시 이웃에 전달함으로서 짧고 안정적인 멀티캐스팅 경로를 형성하도록 한다.

· Flush Tree

그룹 구성 과정에서 노드 i가 노드 j에게 Join Acknowledge를 보냈다면 노드 i는 노드 j의 부모노드가 되고, j는 i의 자식노드가 된다. 만일 일정기간 부모노드에게서 Hello 메시지를 받지 못했다면 멀티캐스팅 경로가 붕괴된 것이므로 자식노드들에게 Flush Tree 메시지를 보내어 경로 붕괴를 알린다. 부모노드로부터 이 메시지를 받은 모든 노드는 자신의 자식노드에게 이 메시지를 다시 전달해야 한다. 참여자 노드가 부모노드로부터 이 메시지를 받으면 다시 Join Request 메시지를 보냄으로서 붕괴된 경로를 재구성하게 된다. MANET의 위상은 시시각각 변화하므로 기존에 형성된 경로를 정기적으로 재구성할 필요가 있다. 또한 페로몬 정보 역시 정기적으로 계산하여 이웃에 전달함으로서 위상 변화를 반영할 수 있도록 해야 한다. 따라서 각 노드는 정기적으로 다음과 같은 연산을 수행하여 그룹을 유지한다.

· Announce_Interval

Core는 매 Announce_Interval마다 Core Announce 메시지를 flooding한다. Core가 아닌 노드는 이 기간이 지났는데도 Core Announce 메시지가 도착하지 않으면 그룹이 사라진 것으로 간주하고 그룹 정보를 삭제한다. 만일 Core가 유일한 송신자가 아니고, 아직 멀티캐스팅 데이터를 받고자 하는 참여자 노드가 있을 경우, 최초로 그룹 정보를 삭제한 참여자 노드는 자신을 Core로 하는 멀티캐스팅 그룹을 재구성한다. 이 때 둘 이상의 노드가 동시에 Core를 선언한 경우, ID가 큰 노드에서 Core 우선권이 있으며, 따라서 ID가 더 작은 노드의 Core Announce 메시지는 무시된다.

· Join_Request_Interval

Core를 제외한 모든 노드는 자신이 Join Request를 보낸 지 Join_Request_Interval 만큼의 시간이 지났으면 메시지를 다시 보낸다. 또한 그룹 내의 모든 노드는 이 시간이 지났는데도 자식 노드로부터 Join Request 메시지를 받지 못했다면 자신의 자식노드 집합에서 삭제한다. forwarding 노드가 이 기간 내에 어떠한 자식노드에게도 Join Request 메시지를 받지 못했다면 forwarding 노드로부터 탈퇴한다.

· Pheromone_Update_Interval

Ad hoc 네트워크 내의 모든 노드는 매 Pheromone_Update_Interval마다 Pheromone Update 메시지를 이웃 노드에 보낸다. 만일 이 기간 내에 자신의 이웃노드로부터 Pheromone Update 메시지를 받지 못했다면 이웃노드 집합에서 삭제한다. 만일 이 이웃노드가 자신의 부모노드라면 그룹의 경로가 붕괴된 것으로 간주하고 자신의 자식노드에게 Flush Tree 메시지를 전달한다.

이에 더하여 이 기간마다 각 노드는 자신의 Ts, Dc, Pc를 계산하고 Pheromone Update 메시지를 통해 이웃노드에 전달한다. 이러한 정기적인 페로몬 전달은 시시각각 변화하는 MANET의 위상을 반영하여 항상 짧고 안정적인 멀티캐스팅 경로를 형성하도록 유도한다.

이제까지는 단일 송신자일 경우의 그룹 구성 및 유지에 대해서 설명하였다. ANMAS는 단일 송신자에 대한 그룹 구성뿐만 아니라 다수 송신자에 대한 그룹 구성을 지원하는데, 이에 대한 설명은 4절에서 다루기로 한다.

4. 다수 송신자에 대한 그룹 구성 및 유지

송신자 기반 트리(source based tree)를 구성하는 경우 수신자들 간의 데이터 패킷 수신율이 균형적이며, ODMRP와 같이 주기적으로 경로를 갱신해줄 경우 효율적인 그룹을 구성할 수 있다. 그러나 송신자의 수가 많을 경우에는 그룹 구성 및 유지에 사용되는 제어 패킷 수가 송신자의 수에 비례하여 늘어나는 문제점이 있다[6].

ANMAS 역시 각 송신자를 Core로 하여 송신자 중심의 그룹을 구성하지만 Pheromone Update 메시지를 공유하여 제어 패킷 수의 증가를 최소화한다. 그룹 내에 둘 이상의 송신자가 있을 경우, 각 송신자는 자신이 Core임을 선언하고 별도의 그룹을 구성한다. 이 경우 각 그룹의 구성 및 유지는 단일 송신자일 경우와 같지만, Pheromone Update 메시지는 모든 Core의 페로몬 정보를 전달한다. Core가 하나인 경우 하나의 Core에 대한 Dc와 Pc 값만을 전달하지만, Core가 여럿일 경우 같은 Pheromone Update 메시지에 여러 Core의 Dc와 Pc를 같이 넣어서 전달하고, Core의 ID를 통해 이를 구분한다.

각 노드는 여러 Core의 Dc, Pc를 저장하고 있다가 Join

Request 메시지를 받았을 때 ID에 따라 구분하여 적절한 연산을 수행한다. 이렇게 Pheromone Update 메시지를 공유함으로서 패킷 전송 부하를 줄이면서도 송신자 중심 트리가 가지는 균형적인 전송율을 유지할 수 있게 된다. ANMAS가 기반으로 하는 CBT 알고리즘은 다양한 장점을 가진 반면, loop가 생길 수 있다는 단점을 가지고 있다[11]. 따라서 이러한 문제점을 해결할 수 있는 부가적인 연산이 필요한데, 이에 대해서는 5절에서 설명하도록 한다.

5. Loop 문제 해결

MANET 상에 적용한 CBT 알고리즘에서의 loop는 멀티캐스팅 효율을 저하시킬 수 있으므로 이러한 문제를 해결할 수 있도록 부가적인 알고리즘을 적용해야만 한다. 본 논문에서는 기존의 OCBT(Ordered Core Based Tree) 알고리즘[11]의 loop 방지 방식을 사용해 이를 해결한다.

먼저 loop가 생기는 경우는 다음과 같다.

경우 1 : 이미 그룹에 참여하고 있는 참여자 노드 둘이 서로에게 Join Request를 보냈다면 이들은 자신이 이미 그룹에 속해 있으므로 Join Acknowledge 메시지를 서로에게 보내게 된다. 그러면 이 두 노드만의 loop가 생성되면서 Core를 포함한 멀티캐스팅 그룹과의 연결이 끊어지게 된다.

경우 2 : 참여자 노드가 보낸 Join Request가 여러 노드를 거쳐 자신의 자식 노드에게로 도착하게 되면 자식 노드는 이미 그룹에 속해 있으므로 참여자 노드에게 Join Acknowledge 메시지를 전달하면서 loop가 생성된다. 참여자 노드가 멀티캐스팅 그룹에 참여하면서 생성되었던 forwarding 노드들은 Join Request 메시지를 받지 못했으므로 그룹에서 탈퇴하게 되고, Core를 포함하는 멀티캐스팅 그룹과의 연결은 끊어지게 된다.

경우 3 : 참여자 노드 i가 다른 참여자 노드 j의 자식 노드에게 Join Request 메시지를 보낸 후, j가 i의 자식노드에게 Join Request 메시지를 보내는 경우이다. 이 때 j는 기존의 경로에 Join Request 메시지를 보내지 않으므로 경로가 붕괴되고, i와 j는 자신의 자식 노드들과 함께 loop에 빠지게 된다.

경우 1, 2, 3에 의해 생성된 loop 내의 노드는 기존의 멀티캐스팅 그룹과 연결이 없으므로 데이터 패킷을 전송하거나 받지 못하게 되고, 이는 그룹의 효율성을 저하시키는 요인이 된다.

Loop 생성을 막는 방법은 다음과 같다. 각 노드는 모두 Lv(level) 정보를 저장하고 있는데 멀티캐스팅 그룹을 구성하기 전에는 모두 ∞으로 초기화된다. 그룹을 시작하는 Core는 Lv를 0으로 초기화한다. Core는 다른 노드의 Join Request를 수락하면서 Join Acknowledge 메시지에 자신의 Lv보다 1이 더 큰 값을 전달한다. Core로부터 Join Acknowledge 메시지를 받은 노드들은 자신의 Lv를 1로 설정한다. 그리고 어떠한 노드도 자신보다 작거나 같은 Lv를 가진 노드의 Join Request 메시지를 수락할 수 없게 하고, 그렇지 않을 경우에는 자신보다 1이 더 큰 Lv를 Join Acknowledge 메시지에 넣어 전달한다. 만약 노드 i가 가진 level, Lv_i가 n일 경우, i에 의해 참여를 수락받은 노드들은 모두 $n+1$의 Lv를 가진다.

그리고 Lv가 n보다 작거나 같은 노드의 Join Request에 대해서는 수락하지 않는다. OCBT의 경우 Core가 가장 높은 Lv를 가지는 반면, 본 알고리즘에서는 0을 가진다는 점이 다르다[10].

이러한 제약을 통해 CBT 알고리즘이 가지는 loop 문제를 해결할 수 있다. 경우 1에서는 두 노드 중 하나의 노드만이 참여를 수락하거나(둘 중 하나의 Lv가 더 클 경우), 둘 모두 참여를 수락하지 않음으로서(둘의 Lv가 같을 경우) loop가 발생시키지 않는다. 경우 2에서 자식노드는 언제나 자신보다 작은 Lv를 가지게 되므로 부모 노드의 Join Request를 수락할 수 없으며, 따라서 loop가 발생할 수 없다. 경우 3에서 모든 노드의 자식 노드는 자신보다 큰 Lv를 가지므로 i 혹은 j에 의해 보내진 Join Request 메시지 중에 적어도 하나는 거부되며, 따라서 loop가 생길 수 없다.

6. 멀티캐스팅 데이터 패킷 전송

ANMAS의 데이터 패킷 전송은 mesh 방식을 따른다. mesh 방식에서는 각 멀티캐스팅 데이터에 대해 순서(sequence)를 붙여 중복된 패킷을 전송하지 않도록 한다. mesh 방식의 멀티캐스팅 데이터 패킷에는 sequence 정보가 포함되어 있는데, 이는 송신자에 의해 정해진다. 최초의 데이터 패킷에는 sequence가 1이고, 송신자는 또 다른 데이터 패킷을 보낼 때마다 이를 1씩 증가시킨다. 또한 각 멀티캐스팅 그룹 노드들은 송신자로부터 받은 데이터 패킷에 담긴 sequence를 저장하고 있다가 같거나 작은 sequence를 가진 데이터 패킷은 무시하고, 그 외의 패킷은 재전송한다. 모든 그룹 노드가 이러한 방식으로 패킷을 보냄으로서 중복되지 않은 데이터 패킷을 모든 그룹 노드가 받을 수 있도록 한다. 또한 데이터 패킷에 송신자의 ID를 포함시켜 자신이 속하지 않는 그룹의 멀티캐스팅 데이터는 무시하도록 한다. mesh 방식이 unicast를 이용하여 데이터 패킷을 보내는 tree 방식보다 유리한 이유는 어떤 노드로부터 받은 데이터 패킷이라 해도 최초로 도착한 데이터 패킷이라면 이를 수용하고 전송한다는데 있다. 즉, unicast로 전달된 패킷은 수신자로 정의된 노드가 아니면 무시해버리기 때문에 경로 붕괴에 취약하다. 하지만 MANET의 broadcast 특성을 이용하는 mesh 방식은 하나의 데이터 패킷만을 받아도 이는 유효한 패킷이 되기 때문에 보다 효율적인 멀티캐스팅을 수행할 수 있다[4].

IV. 성능 평가

1. 시뮬레이션 환경

모든 노드는 고유한 ID를 가지고 있다고 가정하고, 하위 프로토콜이나 트래픽에 의한 패킷 분실은 없다고 가정한다. 2차원 평면 위에 50, 200개의 모빌 노드가 네트워크를 구성하고, 반경 250m의 통신 가능 영역 및 2.0Mbps의 대역폭을 가지며, 0 ~ 20m/sec 속도로 이동한다. 노드의 이동은 난수 목표점 모델(random waypoint model)[12]을 사용한다. 초당 20개의 데이터 패킷을 300초 동안 보내게 되고, 패킷의 크기는 512 byte이다. 난수 목표점 모델의 문제점 때문에[12] 처음 30초간은 데이터 패킷을 보내지 않으므로 총 시뮬레이션 시간은 330초이다. 이는 ODMRP[4]의 실험환경을 따른 것이며, 다만 200개의 노드에 대해 추가로 실험했다는 점이 다르다.

2. 평가 항목 및 실험 결과

멀티캐스팅 효율은 데이터 전송율(delivery ratio), 총 패킷 수

(total packets) 그리고 제어 패킷 수(control packets)로 평가한다.

· 데이터 전송율(%) : 수신자 수를 l, 송신자(source)가 보낸 데이터 패킷수를 x, 수신자가 받은 데이터 패킷의 총 합을 m이라 하였을 때 전송율은 $m / l / x \times 100(\%)$이다. 이는 멀티캐스팅의 효율을 평가하는 가장 중요한 척도이다[6].

· 총 패킷 수 : 참여자 노드 및 forwarding 노드들이 전달한 데이터 패킷들과 MANET 내의 모든 노드들이 그룹의 구성 및 유지를 위해 전달한 패킷들을 모두 합한 것이 총 패킷 수이다. 총 패킷 수가 적다는 것은 같은 일을 하는데 있어서 부하 대비 효율이 좋다는 것을 의미한다.

· 제어 패킷 수 : 그룹의 구성과 유지를 목적으로 전달된 패킷들의 총 합이다. 총 패킷에서 멀티캐스팅 데이터 패킷을 제외한 나머지가 모두 제어 패킷이다. 데이터 패킷 전송이 적을 경우에도 좋은 효율을 발휘하기 위해서는 제어 패킷 수가 적어야 한다.

멀티캐스팅 알고리즘의 성능을 평가하는 항목은 많지만 [6], 대부분의 경우 위의 세 가지 정보만으로도 판단이 가능하다. 다만 전달되는 패킷들의 크기와 관련된 항목들은 패킷 헤더 크기에 대한 정의가 명확하지 않아서 불분명한 값이 되는 경우가 많고, MANET 노드의 broadcast 특성을 고려할 때 패킷의 크기보다는 패킷 전송 횟수가 미치는 영향이 더 크기 때문에 생략하였다.

성능 비교를 위해 선택한 ODMRP 알고리즘은 여러 MANET 멀티캐스팅 알고리즘들[4][5][6][7] 중에서도 단순성, 효율성 확장성을 갖춘 알고리즘으로 알려져 있다[6]. ODMRP는 멀티캐스팅 참여자 중에서 송신자를 중심으로 한 그룹을 구성한다. 송신자는 일정한 시간마다 Join Request 메시지를 flooding하고, 이 메시지를 받은 노드 중에서 송신자가 제공하는 멀티캐스팅 데이터를 받고자 하는 노드는 이 메시지에 대한 Acknowledge 메시지를 보냄으로서 그룹을 구성하게 된다. ODMRP의 Join Request 패킷은 매 1초마다 flooding되도록 설정하였다. 이와 비슷한 제어 패킷 부하를 통해 명확하게 성능을 비교할 수 있도록 ANMAS의 제어 패킷 간격을 다음과 같이 설정하였다.

· Join_Request_Interval : 2초

· Core_Announce_Interval : 20초

· Pheromone_Update_Interval : 1초

두 알고리즘 모두 송신자가 하나일 경우에 대해 실험을 수행하였다.

그림 3 ~ 7은 1000m × 1000m 평면 위에서 50개 노드가 네트워크를 구성할 때 항목별 실험 결과(각각 50회 평균)이다. 이 때 각 변수들의 값은 다음과 같다.

· c = 1.0, α = 0.10, μ = 0.03

그림 8 ~ 12는 2000m × 2000m 평면 위에서 200개의 노드가 네트워크를 구성할 때 항목별 실험 결과(각각 50회 평균)이다. 이 때 각 변수들의 값은 다음과 같다.

· c = 1.0, α = 0.10, μ = 0.07

노드의 수가 200일 때는 50일 경우에 비해 그룹 참여를 위한 경로가 길어지게 되므로 μ를 증가시킴으로서 보다 새로운 이웃에 Join Request를 보내게 하는 것이 효율적이다.

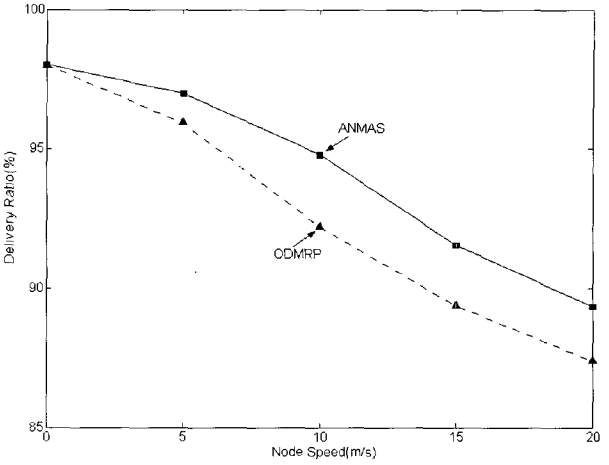


그림 3. 참여자 수가 2일 때 전송율(%).

Fig. 3. Packet delivery ratio (2 members).

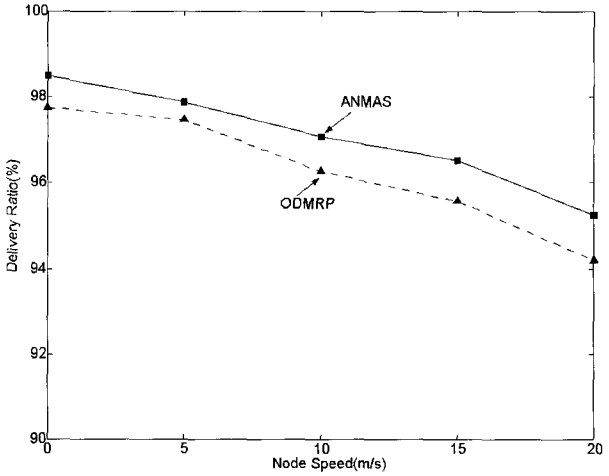


그림 4. 참여자 수가 5일 때 전송율(%).

Fig. 4. Packet delivery ratio (5 members).

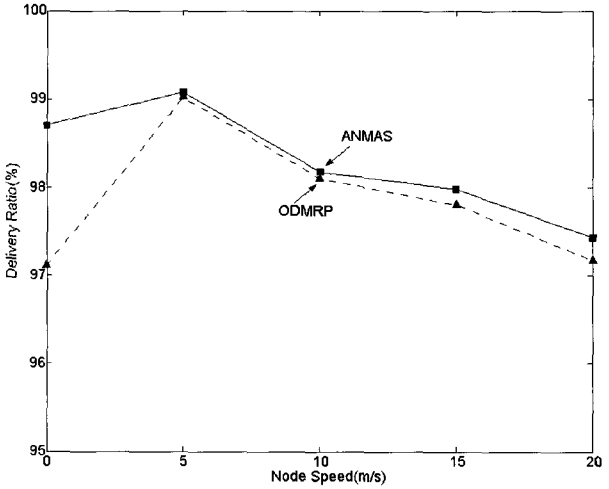


그림 5. 참여자 수가 10일 때 전송율(%).

Fig. 5. Packet delivery ratio (10 members).

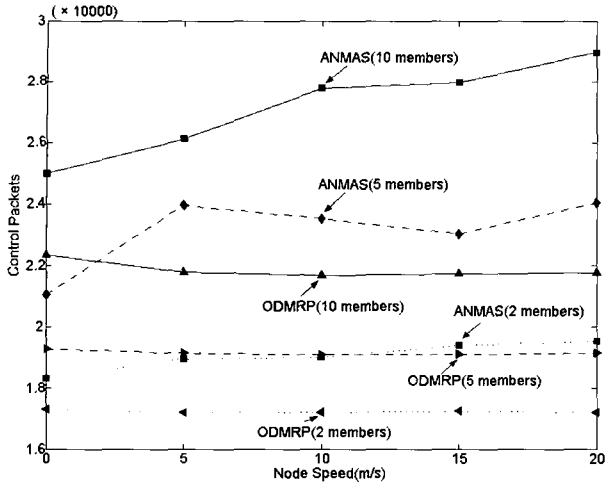


그림 6. 제어 패킷 수(50 노드).
Fig. 6. Control packets(50 nodes).

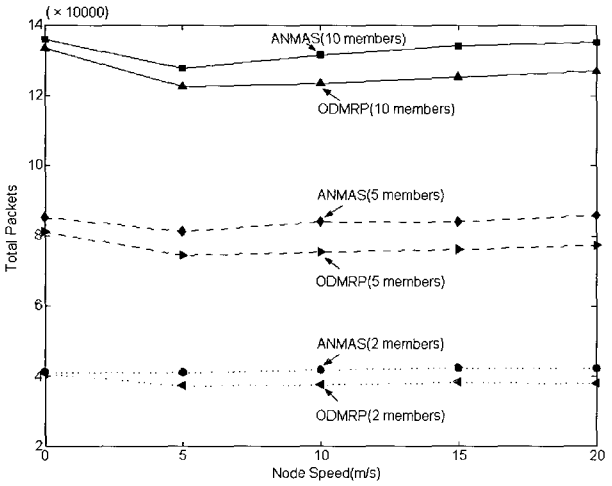


그림 7. 총 패킷 수(50 노드).
Fig. 7. Total packets(50 nodes).

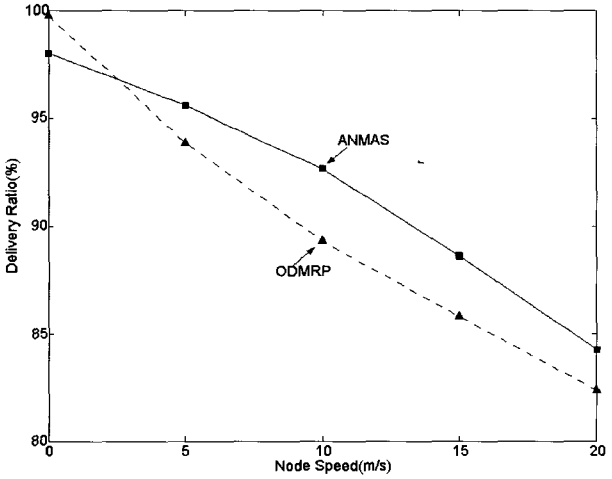


그림 8. 참여자 수가 5일 때 전송율(%).
Fig. 8. Packet delivery ratio(5 members).

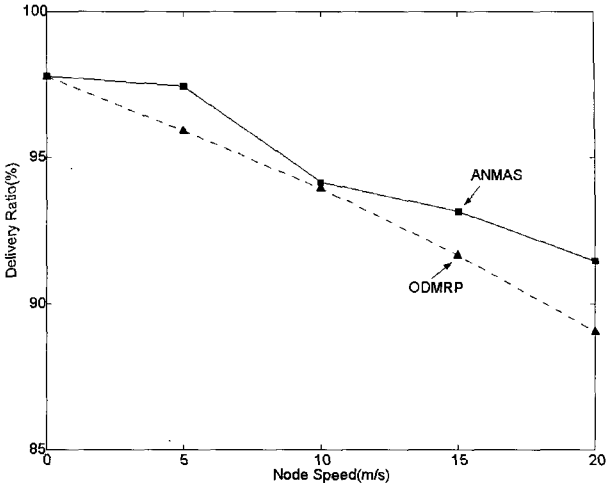


그림 9. 참여자 수가 10일 때 전송율(%).
Fig. 9. Packet delivery ratio(10 members).

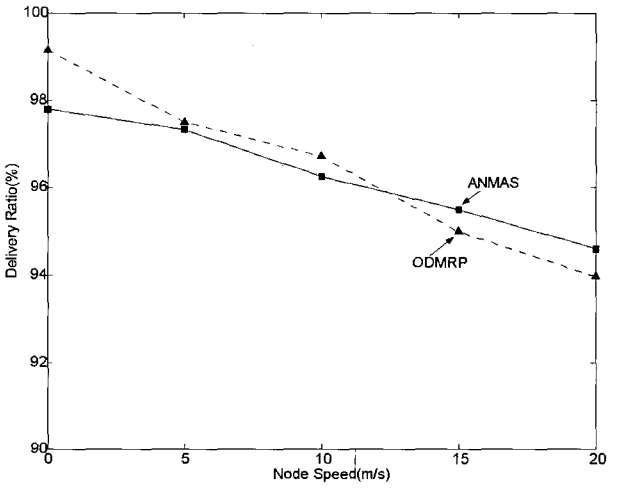


그림 10. 참여자 수가 20일 때 전송율(%).
Fig. 10. Packet delivery ratio(20 members).

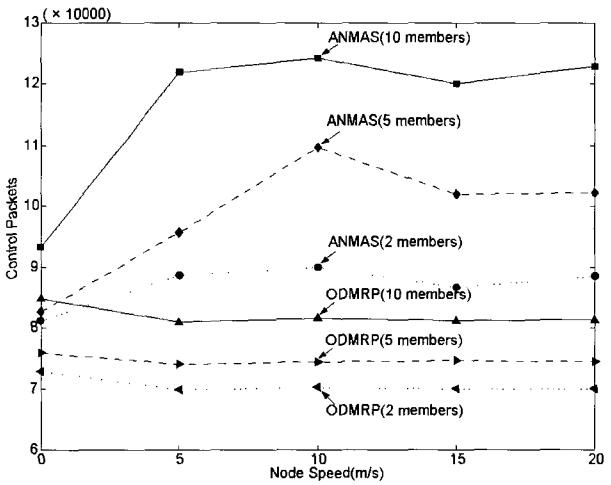


그림 11. 제어 패킷 수(200 노드).
Fig. 11. Control packets(200 nodes).

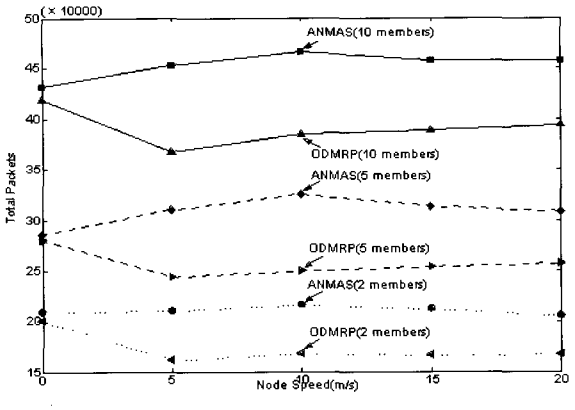


그림 12. 총 패킷 수(200 노드).
Fig. 12. Total packets(200 nodes).

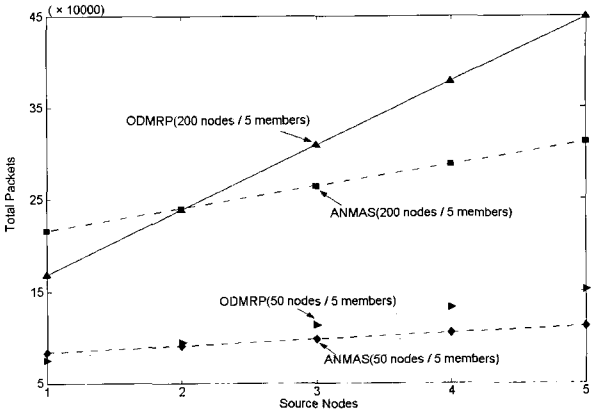


그림 13. 송신자 수에 따른 총 패킷 수.
Fig. 13. Number of total packets increased by source nodes.

표 1, 2, 3은 1000m × 1000m 평면 상에 50 노드가 있고, 참여자 수가 5, 그리고 최대 20 m/s로 움직일 경우에 α, μ에 따른 전송율 및 총 패킷 수의 차이를 보여준다.

효율적인 α, μ 값은 노드 수나 참여자 수에 따라 다르다. 다만 표 2, 3에 의하면 전반적으로 α, μ가 증가할수록 패킷 수 역시 증가하면서 패킷 수 대비 전송율은 떨어짐을 알 수 있다. 따라서 효율적인 멀티캐스팅 그룹을 구성하기 위해서는 α, μ 모두 [0.01, 0.15] 구간의 값을 가지는 것이 적당하다. 그림 13은 송신자 수의 증가에 따라 각 알고리즘의 총 패킷 수가 어떻게 증가하게 되는지를 보여주는 그래프이다.

표 1. α, μ에 따른 전송율 차이.
Table1. Delivery ratio depend on parameter α, μ.

μ	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.10$	$\alpha = 0.15$
0.01	94.12	94.96	95.00	94.40
0.05	95.03	95.13	94.68	94.94
0.10	94.86	94.90	95.21	94.97
0.15	95.44	95.56	95.13	95.56

표 2. α, μ에 따른 총 패킷 수 차이.
Table2. Total packets depend on parameter α, μ.

μ	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.10$	$\alpha = 0.15$
0.01	84099	83864	85206	88176
0.05	84367	84211	86105	87428
0.10	86166	86513	87640	89356
0.15	85331	88345	89948	90732

표 3. 전송율 / 총 패킷 수(%).
Table3. Delivery ratio / total packets(%).

μ	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.10$	$\alpha = 0.15$
0.01	0.111916	0.113231	0.111494	0.107059
0.05	0.112639	0.112966	0.109959	0.108546
0.10	0.110090	0.109694	0.108638	0.106283
0.15	0.111847	0.108167	0.105761	0.105321

노드 속도는 10m/s이고 총 노드 수가 50, 200일 때 5개의 참여자 노드가 있을 경우이다.

3. 시뮬레이션 결과 분석

3.1 송신자 수가 1일 경우

그림 3, 4, 8, 9에서와 같이 참여자 수가 적은 경우 전송율 면에서 ANMAS가 ODMRP에 비해 훨씬 나은 성능을 보인다. 그림 5, 10과 같이 참여자 수가 많아지는 경우에는 ANMAS가 조금 낫거나 비슷한 성능을 보임을 알 수 있다. 참여자 수가 많을 경우에는 forwarding 노드를 포함한 그룹 내 노드 수가 증가하고, 이들이 여분의 경로를 형성하면서 전반적으로 패킷 전송율이 향상되기 때문이다. 이 점은 mesh 방식의 멀티캐스팅 알고리즘이 가진 장점이다. 참여자 수를 더 증가시켰을 때에 두 알고리즘의 전송율은 거의 비슷하였다. 그림 6, 11의 제어 패킷 수, 그리고 그림 7, 12의 총 패킷 수에서 볼 수 있듯이 제어 패킷 및 총 패킷 수에 있어서는 ANMAS가 ODMRP에 비해 더 많다. 이는 ANMAS가 경로 붕괴시 재구성을 위한 제어 패킷을 보내고, 안정성을 위해 더 많은 forwarding 노드를 포함하기 때문이다. 하지만 두 알고리즘의 전송율을 고려해볼 때 효율면에서 크게 뒤지지 않는다고 볼 수 있다.

3.2 송신자 수가 2 이상일 경우

각 알고리즘은 송신자 중심의 그룹을 구성하므로 송신자 수에 따라 전송율이 변하지 않는다. 따라서 송신자 수가 증가하여도 ANMAS는 ODMRP에 비해 나은 전송율을 보인다.

그림 13을 보면 송신자 수의 증가에 따른 총 패킷 수의 증가는 ODMRP가 훨씬 빠르고, 송신자 수가 2 이상일 경우, 즉 다수 대 다수의 멀티캐스팅을 수행할 경우에 총 패킷 수는 ODMRP가 더 많음을 알 수 있다. 실험 환경에서 Pheromone Update 패킷의 수는 50 노드일 때 약 16,500개, 200일 때 66,000개 정도인데, 그림 6, 11과 비교해보면 이 비율이 제어 패킷의 50% ~ 90%에 해당함을 알 수 있다. Pheromone Update 패킷의 수는 송신자 수에 따라 증가하지 않고 일정하기 때문에 ANMAS의 제어 패킷 수 증가가 ODMRP보다 훨씬 적다. 따라서 송신자 수가 2 이상일 때에는 전송율 면에서나 패킷 부하면에서나 ANMAS가 더 우수하다는 것을 알 수 있다.

실험 결과를 종합적으로 분석해볼 때 ANMAS가 더 많은

패킷을 사용한다. 하지만 전송율에 있어서는 전체적으로 더 나은 면을 보인다. 특히 ODMRP 알고리즘은 송신자 수에 따라 제어 패킷수가 크게 증가지만 ANMAS는 Pheromone Update 메시지를 공유함으로서 적은 제어 패킷 수 증가만으로도 뛰어난 전송율을 보인다. 따라서 무선 회의, 재난 구조와 같은 다수 대 다수의 멀티캐스팅이 필요한 응용분야에서 더 유용한 알고리즘이 될 수 있다.

V. 결론 및 추후과제

ANMAS는 mesh 방식의 MANET 멀티캐스팅 알고리즘으로서 개미 알고리즘을 통해 얻은 Core의 위상정보를 이용, 안정적인 경로를 형성하고, 경로 붕괴에 대처함으로서 기존의 ODMRP 알고리즘에 비해 높은 전송율을 얻을 수 있었다. 특히 개미 알고리즘의 간접적인 정보 전달방식을 응용하여 송신자 중심 트리의 이점을 살리면서도 다수의 송신자에 의해 발생하는 제어 패킷 증가율을 감소시킴으로서 다수 대 다수의 멀티캐스팅에 있어서 월등한 효율을 보인다. 또한 노드의 수가 증가하고 통신 영역이 넓은 경우에도 좋은 전송율을 보임으로서 네트워크 크기에 따른 확장성 면에서도 뛰어남을 보여준다. 향후에는 개미 알고리즘에서 사용하는 다양한 페로몬 부여 및 제어 방식을 ANMAS에 적용해봄으로서 성능 향상을 도모하고자 한다. 또한 ANMAS의 특성을 이용하여 MANET 상의 라우팅 문제를 해결하거나, 근래에 연구가 활발한 계층적 멀티캐스팅 알고리즘을 적용하여 한층 발전된 형태의 멀티캐스팅 알고리즘을 개발하는 것도 고려할만 하다.

참고문헌

[1] A. Ballardie J. Crowcroft and P. Francis, "Core based trees(CBT) An Architecture for Scalable Inter-Domain Multicast Routing," *In Proceedings of ACM SIGCOMM '93,* pp. 85-95, 1993.

[2] M. Abolhasan, T. Wysocki, E. Dutkiewicz, "A review of routing protocols for mobile Ad hoc networks," Ad Hoc *Networks 2,* pp. 1-22, 2004.

[3] M. Dorigo and G. Di Caro, "The ant colony optimization metaheuristic," *New Ideas in Optimization,* pp. 11-32, 1999.

[4] S. Lee, M. Gerla and C. Chiang, "On-demand multicast routing protocol," *In Proceedings of IEEE WCNC'99,* pp. 1298-1302, 1999.

[5] C. W. Wu, Y. C. Tay, "AMRIS: a multicast protocol for Ad hoc wireless newtorks," *In Proceedings of IEEE MILCOM'99,* 1999.

[6] S. Lee, W. Su, J. Hsu, M. Gerla and R. Bagrodia, "A performance comparison study of Ad hoc wireless multicast protocols," *In Proceeding of IEEE INFOCOM'00,* 2000.

[7] J. J. Garcia-Luna-Aceves and E. L. Madruga, "The core-assisted mesh protocol," *IEEE Journal on Selected Areas in Communications,* vol. 17, no. 8, pp. 1380-1394, 1999.

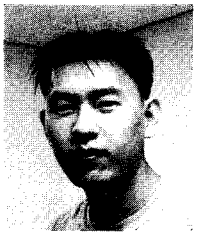
[8] C. Shen and C. Jaikaeo, "Ad hoc multicast routing algorithm with swarm intellegence," *Technical Report 2003-08,* University of Delaware, 2003.

[9] R. C. Eberhart and J. Kennedy, Swarm Intelligence, Morgan Kaufmann, San Francisco, 2001.

[10] S. Lee, W. Su, and M. Gerla, "Ad hoc wireless multicast with mobility prediction," *In Proceeding of IEEE ICCCN'99,* pp. 4-9, 1999.

[11] C. Shields, J. J. Garcia-Luna-Aceves, "The ordered core based tree protocol," *In Proceeding of IEEE INFOCOM'97,* 1997.

[12] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," *In Proceedings of IEEE INFOCOM'03,* 2003.

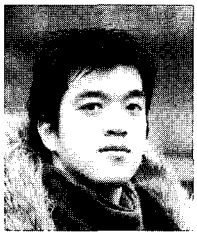


이 세 영

2004년 서강대 컴퓨터학과 졸업. 2004년~현재 동대학 석사과정 재학. 관심분야는 최적화 알고리즘, 무선 네트워크 등.

장 형 수

1994년 미국 Purdue University, 전기및 컴퓨터공학과 졸업, 동대학 석사, 박사 (1996, 2001). 2001년 9월 ~ 2002년 6월 Institute for Systems Research (ISR), University of Maryland, College Park, Postdoc, 2002년 6월 ~ 2003년 2월 고려대학교 정보통신 기술 공동연구소 연구교수, 2003년 3월 ~ 현재 서강대학교 공과대학 컴퓨터학과 조교수.



김 중 항

2004년 서강대 컴퓨터학과 졸업. 2004년~현재 동대학 석사과정 재학. 관심분야는 최적화 알고리즘, 센서 네트워크, Peer to Peer (P2P) 등.