Ad hoc 네트워크에서 제어메시지 부하를 감소시키는 클러스터 유지 방법

(A Cluster Maintenance Scheme to Reduce the Control Overhead in Mobile Ad hoc Networks)

왕 기 철 † 방 상 원 †† 조 기 환 †††
(Gi-Cheol Wang) (Sang-Won Bang) (Gi-Hwan Cho)

요 약 클러스터 구조는 Ad hoc 네트워크내의 전체 호스트로 데이타를 방송하는 경우에, 재전송되는 메시지의 수를 감소시킨다. 이러한 클러스터 구조의 이점을 보존하기 위해 클러스터 유지방법이 이용된다. 그러나 기존의 클러스터 유지방법들은 이웃정보 파악을 위한 제어메시지 외에도 클러스터 재구성을 위한 추가적인 메시지 교환을 필요로 한다. 이로 인해 클러스터 구조의 유지에 따른 이점은 크게 약화된다. 본 논문에서는 클러스터 구조의 중첩성을 이용하여 Hello시간에 클러스터 헤드들만 제어메시지를 broadcast 전송하고 일부 멤버 호스트들은 제어메시지의 unicast전송을 통해 분리된 게이트웨이를 파악하는 클러스터 유지방법을 제안한다. 제안하는 방법은 클러스터 재구성이 필요할 때에도, 각 호스트간에 전송되는 제어메시지를 최소한으로 줄이기 위한 전략을 사용한다. 제안된 방법은 이 과정에서 2홉 클러스터의 정의를 파괴하지 않으며, 클러스터를 완전히 분산된 방법으로 생성한다. 본 논문에서 제안한 방법은 실험결과에 의해 LCC[1]보다 우수한 것으로 평가된다.

키워드 : Ad hoc 네트워크, 클러스터 구조, 제어메시지 부하, 클러스터 유지방법

Abstract The cluster structure reduces the number of retransmission messages, when a broadcast to all hosts in ad hoc network is needed. A cluster maintenance scheme is employed to preserve this advantage from time to time. However, most of the cluster maintenance schemes require additional control messages for cluster reformation as well as control messages for acquiring neighbor information. This mitigates the advantages of employing cluster structure in ad hoc network. In this paper, a cluster maintenance scheme which forces only clusterheads to broadcast control messages during hello time is proposed. When the cluster reformation is needed, the proposed scheme employs a strategy to reduce the control messages to a minimum. In these processes, the proposed scheme doesn't violate the definition of 2-cluster and produces the clusters in fully distributed method. The simulation results prove that our scheme is better than LCC[1].

Key words : Ad hoc network, cluster structure, control message overhead, cluster maintenance

1. 서 론

Ad hoc 네트워크는 임의의 이동호스트들이 기반 통신시설(infrastructure)을 이용할 수 없는 경우에 즉석의 통신을 수행하기 위하여, 다른 이동호스트의 도움을 받아 통신을 수행하는 네트워크이다. 따라서 이동 Ad hoc 네트워크에서 각 이동 호스트들은 라우터로서의 기능을 수행하여야 하고, 임의의 통신경로는 다중의 홉으로 구성될 수 있다.

Ad hoc 네트워크에서 이동호스트들을 그룹으로 묶어 관리하면 그룹단위로 채널을 관리할 수 있고, 제어메시지의 교환부하 감소 및 이동성 관리가 용이해진다[2]. 또한 임의의 Ad hoc 네트워크를 클러스터 헤드들간의 네트워크로 만들어 네트워크의 확장성(scalability)을 증가시킨다.

이동 호스트들을 일단의 기준에 따라 묶은 그룹을 클러스터(cluster)라 하고 이렇게 이동 호스트들을 그룹화하는 것을 클러스터 구성(cluster formation)이라 한다.

† 학생회원 : 전북대학교 컴퓨터통계정보학과
gcwang@dcs.chonbuk.ac.kr
†† 비 회 원 : 송원대학 컴퓨터정보과 교수
swbang@songwon.ac.kr
††† 종신회원 : 전북대학교 전자정보공학부 교수
ghcho@dcs.chonbuk.ac.kr
논문접수 : 2003년 7월 21일
심사완료 : 2003년 11월 6일

또한 임의의 이동호스트가 최소 하나 이상의 클러스터에 포함되는 구조를 클러스터 구조라 한다. 이때 클러스터 내에서 호스트들간의 최대 경로길이에 따라 1홉 클러스터(clique), 2홉 클러스터, 4홉 클러스터 등으로 구분된다. 1홉 클러스터와 4홉 클러스터는 특수한 구조로 클러스터 구조를 유지·보수하는데 많은 통신비용이 소모된다. 따라서 본 논문에서는 가장 일반적인 2홉 클러스터를 가정한다.

2홉 클러스터의 구조는 클러스터 헤드가 모든 멤버들과 직접통신이 가능하도록 연결되어 있다. 따라서 클러스터 헤드가 데이타를 방송하면 그 클러스터내의 모든 호스트들이 동시에 수신할 수 있다. 그러므로 이러한 특성은 reactive 방식의 라우팅 프로토콜(DSR[3], AODV[4])이나 CBRP[5]와 같은 클러스터헤드 기반의 라우팅 프로토콜에서 경로요청(route request)동안 메시지의 재전송 횟수를 줄이는데 유용하게 이용될 수 있다. 그러나 이러한 클러스터 구조를 유지하기 위해서는 클러스터 유지를 위한 전략이 요구되고, 이러한 전략들은 많은 추가적인 통신 오버헤드를 유발한다. 이러한 통신 오버헤드는 이웃정보 파악을 위한 제어메시지와 클러스터 재구성을 위한 메시지들을 포함한다.

본 논문에서는 이 두 종류의 제어메시지를 모두 줄이는 클러스터 유지 방법을 제안한다. 먼저 이웃정보 파악을 위한 제어메시지의 감소는 클러스터 구조를 이용한 방송 절차에서 단지 게이트웨이 호스트들만 인접 클러스터 헤드들을 인지할 필요가 있다는 사실에 근거한다. 즉 클러스터 헤드들은 항상 메시지를 방송하는 지역방송자(local broadcaster)역할을 수행하므로, 이웃의 보통 호스트에 대해 인지할 필요가 없다. 반면에 게이트웨이 호스트들은 임의의 방송메시지를 수신한 후 다음 클러스터 헤드에 unicast 전송하므로, 인접한 클러스터 헤드들을 인지해야 한다. 즉, Hello 메시지 교환시간에 단지 클러스터 헤드들만 제어메시지를 방송하도록 하자는 것이다. 그리고 새로 생성된 게이트웨이 호스트들간의 연결을 인지하기 위해서 단일 클러스터에만 소속되는 보통 호스트들이 클러스터 헤드의 제어메시지에 unicast로 응답한다. 이때 다른 클러스터에 속한 보통 호스트는 promiscuous 모드로 동작하여 분리된 게이트웨이 호스트를 인지한다.

반면에 클러스터 재구성을 위한 메시지를 감소시키기 위해서는 기존의 weight 기반의 클러스터 유지방법을 수정한다. 즉, 클러스터 헤드들끼리 인접하였을 경우에는 많은 클러스터 헤드를 가지는 호스트가 멤버 호스트로 역할을 변경한다. 이 경우에 멤버호스트로 역할을 변경하는 호스트들만 메시지를 방송하게 된다. 두 번째로 클러스터 헤드가 유실된 이동 호스트들은 (자신의 id. × 임의의 시간상수)만큼을 대기 한 후, 먼저 클러스터 헤드임을 선언한 호스트가 클러스터 헤드가 된다. 즉, 자신이 클러스터 헤드임을 선언하기 전에 이웃으로부터 클러스터 헤드 선언 메시지를 수신하는 호스트는 메시지를 방송하지 않게 된다.

본 논문은 다음과 같이 구성된다. 2장에서는 클러스터 유지 방법에 관한 기존연구들에 대해 간략히 기술한다. 3장에서는 본 논문에서 제안하는 클러스터 유지방법을 기술한다. 4장에서는 제안하는 방법과 LCC를 비교한 실험 결과를 보이고 5장에서는 결론을 내린다.

2. 관련 연구

일반적으로 Ad hoc 네트워크에서 클러스터 멤버쉽에 변화가 있을 때마다 클러스터를 다시 구성하지 않고 클러스터 유지방법을 이용한다면 클러스터의 변화수는 크게 감소될 수 있다고 알려져 있다. LCC(Least Clusterhead Change)[1]는 잦은 클러스터 헤드의 변화를 줄이기 위해, 클러스터 재구성의 영역을 최소한으로 줄이는 방법을 사용하였다. 즉, 이 방법은 임의의 호스트 자신 또는 이웃 호스트들의 이동으로 인해, 클러스터 헤드가 유실된 호스트들이나 클러스터 헤드끼리 경합을 벌이는 영역에 속한 호스트들만 클러스터 구성을 다시 하도록 한다. 그러나 이 방법은 클러스터내의 멤버쉽 변화를 체크하기 위해, 주기적으로 각 이동 호스트가 이웃정보 획득을 위한 제어 메시지를 교환해야 하고, 클러스터 재구성을 위한 제어 메시지 교환을 추가적으로 수행해야 한다. 대부분의 클러스터 유지 방법[6-8]들은 LCC와 유사한 클러스터 유지 방법을 사용한다.

ACM(Adaptive Cluster Maintenance)은 클러스터 헤드 없이 각 이동 호스트들이 자신의 지역성(2홉이내)과 연결성에 기반하여 클러스터 멤버쉽 수정을 하는 방법을 제안하였다. 이 방법은 클러스터의 변화를 줄이는 이점이 있으나, 이웃파악을 위한 제어메시지는 물론이고 2홉 내의 지역성과 연결성을 고려하는 제어메시지를 추가적으로 교환해야 하므로 통신 오버헤드가 크다. 참고문헌 [8]에서는 클러스터 헤드를 가지는 4홉 클러스터 구조에서의 클러스터 유지방법을 제안하였다.

ABCM(Access Based Cluster Maintenance)[9]는 클러스터 구성 및 유지를 위한 임의의 공통채널을 하나 설정한 후에 먼저 이 채널을 접근하는 호스트가 클러스터 헤드가 되고, 나중에 접근하는 호스트는 이 채널을 통해 멤버 노드가 되게 하는 링크 계층 기반의 클러스터 구성 방법을 사용한다. 이 클러스터 유지방법은 각 클러스터 헤드가 Hello 메시지를 주기적으로 혹은 보통 호스트의 요청으로 전송하고 보통 호스트들은 가입메시지를 전송하여 클러스터 멤버쉽을 유지한다. 특정기간

동안 Hello메시지를 수신하지 못하는 보통 호스트들은 클러스터 헤드가 되며 클러스터 헤드들은 자신의 클러스터를 떠나기 이전에 분리 메시지를 전송하여 잔류 호스트들 중 하나가 클러스터 헤드가 되도록 한다. 이 방법은 병합(merge)방법을 사용한다 하더라도, 많은 수의 클러스터를 생성하여, 클러스터 구조의 이점을 약화시킨다. 참고문헌 [10]에서는 클러스터 구성 및 유지를 위한 제어메시지를 교환하지 않고, MAC계층에서 라우팅을 위해 전송되는 제어패킷(HELLO, RREQ, RREP, RERR)에 클러스터 상태 정보를 삽입하여 클러스터 구조를 유지하는 방법을 제안하였다.

대부분의 기존 클러스터 유지 방법들은 클러스터의 변화를 작게 하기 위하여 필요한 weight(id., connectivity) 값에 기반하여 클러스터 구조를 유지하였다. 그러나 이러한 weight기반 방법은 주기적인 Hello 메시지 교환은 물론 클러스터 재구성을 위한 추가적인 제어메시지의 교환을 요구한다. 따라서 클러스터 구조를 사용함으로써 얻는 방송메시지수의 감소효과를 크게 약화시킨다. 이것을 방지하기 위해서는 weight기반의 클러스터 구조 유지전략을 수정할 필요성이 있다. 본 논문에서는 클러스터 구조의 중첩성에 기반하여 방송되는 제어메시지의 수를 감소시키는 방법을 제안한다. 또한 클러스터 구조의 유지를 위해, 클러스터들 간의 경합상황에서는 인접 클러스터헤드의 수에 따라, 멤버호스트들 사이의 경합에서는 선언순서에 따라 클러스터헤드를 선출하는 방법을 제안한다.

3. 제어메시지 부하를 감소하는 클러스터유지 방법

클러스터 구조를 이용한 메시지의 방송은 메시지를 전송하는 호스트의 종류에 따라 다른 형태로 전송된다. 즉, 클러스터 헤드는 메시지를 broadcast 전송하지만, 이를 수신하는 멤버 호스트들은 인접한 다른 클러스터 헤드나 인접 클러스터의 게이트웨이 호스트에게 unicast 전송한다. 이러한 이유로 클러스터 구조에서 멤버 호스트들만 이웃의 클러스터헤드나 게이트웨이 호스트의 존재를 인지할 필요가 있다. 따라서 클러스터 구조를 유지한다면, Hello 메시지 전송시간에 클러스터 헤드들만 Hello 메시지를 방송할 필요가 있다. 그러나 이 경우에 이웃 클러스터의 게이트웨이 호스트에 대해서는 인지하지 못한다. 예를들어 그림 1에서 보는 것처럼 호스트 17과 18은 각각 자신의 클러스터 헤드 3과 15를 인지하지만, 각각의 호스트가 상대방의 게이트웨이 역할을 수행해야 함을 인지할 수가 없다. 가장 단순한 해결책은 Hello 메시지를 수신한 모든 호스트가 클러스터 헤드에게 자신의 클러스터 가입 메시지로 응답하면 된다. 그러나 이 방법은 많은 제어메시지 부하를 유발한다.

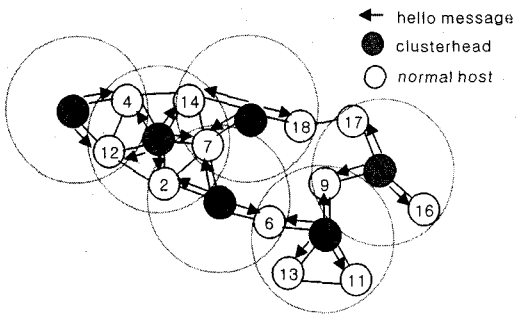


그림 1 클러스터 구조에서의 클러스터 헤드들의 hello 메시지 방송

본 논문에서는 Hello시간에 하나의 클러스터에만 속하는 멤버 호스트들이 unicast로 응답하여 인접한 클러스터의 게이트웨이 호스트를 인지하는 방법을 제안한다. 이 경우에 각 이동 호스트는 promiscuous 모드[3]로 운영되는 것을 전제로 한다. 그림 1에서 하나의 클러스터에만 소속되는 멤버 호스트들은 호스트 11, 13, 16, 17, 18이다. 이들이 각각 자신의 클러스터 헤드에 응답한다면, 호스트 17과 호스트 18은 각각 인접 클러스터의 게이트웨이인 상대방을 인지하게 된다. 참고문헌 [11]과 [12]에서는 RREQ와 같은 flooding되는 메시지를 이용하여, Hello 패킷을 감소시키는 시도를 하였다. 그러나 이 기법은 flooding되는 메시지에 의존하므로, flooding에 의한 충돌 및 메시지부하를 증가시키고 flooding이 자주 발생하지 않으면 효율이 떨어진다. 반면에 본 논문에서 제안하는 방법은 클러스터 구조의 특성을 이용하여 제어메시지를 감소시키므로, flooding의 빈도수에 상관없이 제어 메시지를 감소시킬수 있다. 더구나 제안하는 방법은 클러스터 구조를 이용함으로써 재전송되는 메시지수는 물론 경합 및 충돌도 크게 감소시킬수 있다.

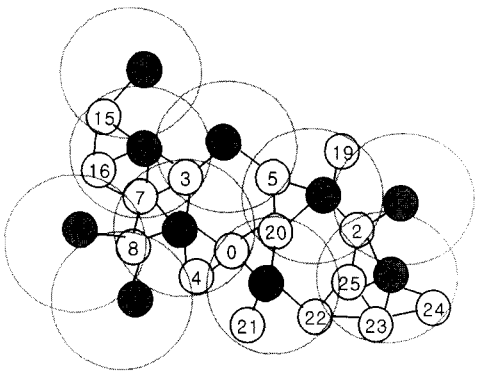
클러스터 구조는 위에서 언급한 여러 가지 이점을 제공해주지만, 이러한 이점을 보존하기 위해서는 지속적으로 클러스터 구조를 유지해야 한다. 그러나 클러스터 구조를 유지하기 위해서는 Hello시간에 교환되는 제어 메시지 외에도 클러스터 재구성을 위한 추가적인 메시지 교환이 필요하다. 이는 호스트들의 이동에 의한 클러스터 헤드로부터의 분리 및 단일 클러스터에서 다중 클러스터 헤드의 경합이 발생하기 때문이다. 이러한 추가적인 메시지는 클러스터 구조를 이용함으로써 얻는 재전송 메시지의 부하감소 효과를 크게 약화시킨다. 이는 기존의 클러스터 유지방법들이 대부분 임의의 weight(id., connectivity)에 대한 비교를 수행하여, 클러스터 변화를 최소화하고자 했기 때문이다. 따라서 이러한 제어 메

시지들을 감소시키기 위해서는 이러한 weight기반 전략을 수정하는 클러스터 유지방법을 사용할 필요가 있다.

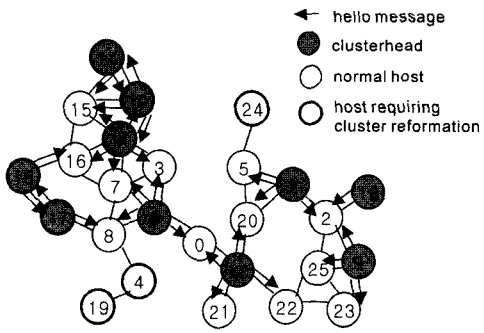
본 논문에서는 다음과 같은 클러스터 유지 방법을 이용하여 클러스터 재구성에 따른 제어 메시지 교환 부하를 감소시킨다.

1. 클러스터 헤드였던 호스트가 다른 클러스터 헤드로부터 Hello 메시지를 수신하였다면, 그 클러스터 헤드들에게 {인접 클러스터 헤드수, 자신의 id}를 unicast 전송한다. 각 클러스터헤드는 이웃으로부터 수신한 이 정보를 순서대로 비교하여 자신이 승자인지 아닌지를 판단하게 된다. 즉 인접 클러스터 헤드수가 이웃 클러스터 헤드보다 작은 호스트가 승자가 되고, 만일 같은 경우에는 id.가 작은 호스트가 승자가 된다. 경합에서 패자가 된 호스트들은 클러스터 헤드 포기 메시지를 방송한다.
2. 임의의 시간 동안 어떠한 클러스터 헤드로부터도 Hello 메시지를 수신하지 못한 보통 호스트 혹은 클러스터 헤드 포기 메시지로 인해 클러스터 헤드가 없어진 보통 호스트들은 먼저 클러스터 헤드임을 선언한 호스트가 클러스터 헤드가 되는 방식으로 클러스터를 재구성 한다. 이때 이웃의 호스트간에 서로 클러스터 헤드선언 하는 것을 방지하기 위해, 자신의 id $\times \alpha$(선언 대기 시간)만큼 대기한 후, 클러스터 헤드임을 선언한다. 즉, 자신의 대기시간 내에 다른 클러스터 헤드로부터 선언메시지를 수신하는 호스트는 그 호스트를 자신의 클러스터 헤드로 등록한다.
3. 단일 클러스터에만 소속된 호스트들은 자신의 클러스터 헤드에게 가입메시지를 unicast전송한다.

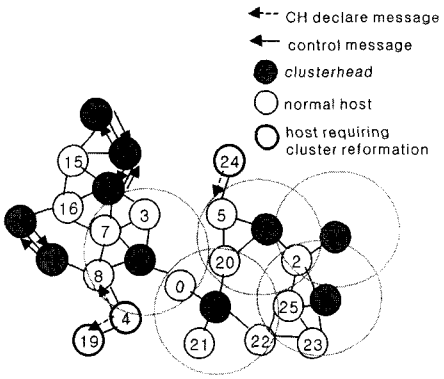
제안하는 클러스터 유지방법의 이해를 돕기 위해, 도식적인 예제를 도입해보자. 그림 2의 (a)는 임의의 시점에서 클러스터 구성된 네트워크를 나타내고 (b)는 다음 Hello 시간에 클러스터 헤드들에 의한 Hello 메시지의



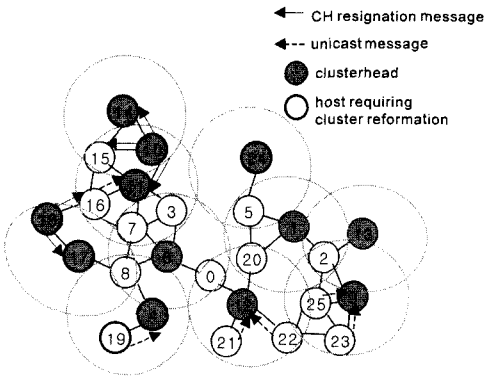
(a) 클러스터 구성된 ad hoc 네트워크



(b) 다음 hello 시간



(c) 클러스터헤드의 재설정



(d) 클러스터헤드 포기메시지 및 단일 클러스터 소속 멤버의 unicast전송

그림 2 임의의 Hello시간 이후의 제안된 방법의 클러스터 재구성 과정

방송 및 클러스터 재구성이 필요한 호스트들을 보여준다.

그림 2의 (b)에서 호스트 10, 11, 14, 17, 18은 이웃에 다른 클러스터 헤드가 존재하기 때문에 클러스터 재구성이 필요하고, 호스트 4, 19, 24는 클러스터 헤드가 존재하지 않기 때문에 클러스터 재구성이 필요하다. 이때 클러스터 헤드끼리 인접한 호스트 10, 11, 14, 17, 18은 각각 그림 2의 (c)에서 보는 것처럼, {인접 클러스터 헤드수, 자신의 id.}를 unicast 전송한다.

한편, Hello 시간동안 클러스터 헤드로부터 Hello 메시지를 수신하지 못한 호스트 4, 19, 24는 각 호스트마다 다른 대기시간을 거친 후, 자신이 클러스터 헤드임을 선언한다. 일반적으로 id.가 작은 노드는 대기시간이 짧게 되므로, 먼저 클러스터 헤드임을 선언하게 된다. 즉, 호스트 4는 19보다 먼저 클러스터 헤드임을 선언하게 되고, 호스트 19는 이 클러스터에 편입된다. 반면에 호스트 24는 경합 호스트가 존재하지 않으므로 스스로 클러스터 헤드가 된다.

이웃 클러스터 헤드의 기준값들과 비교해서 자신이 패자가 된 호스트는 더 이상 클러스터 헤드가 아니라는 사실을 이웃들에게 방송한다. 그림 2의 (d)에서 패자 호스트 10은 호스트 11, 14를 클러스터 헤드로 설정하고 패자 호스트 18은 호스트 17을 클러스터 헤드로 설정한다. 그리고 이 사실을 이웃 호스트들에게 클러스터 가입 메시지를 통해 방송한다.

마지막으로 단일 클러스터에만 소속된 보통 호스트 16, 19, 21, 22, 23은 자신의 클러스터 헤드에게 클러스터 가입메시지를 unicast전송한다. 이때 각각의 호스트는 promiscuous모드로 운영되기 때문에 분리된 게이트웨이들을 인지하게 된다. 즉, 호스트 22의 가입 메시지 전송시 인접한 호스트 25와 23은 호스트 22가 분리된 게이트웨이 임을 인지하게 된다. 같은 방법으로 호스트 22는 호스트 23과 호스트 25를 그리고 호스트 18은 호스트 16을 인지하게 된다.

본 논문에서 제안하는 클러스터 유지방법은 클러스터끼리 인접한 경우 특정한 기준에 의거 클러스터 헤드임을 포기하는 호스트가 반드시 존재하게 되므로, 2홉 클러스터의 정의를 파괴하지 않는다. 또한 클러스터 헤드가 존재하지 않는 경우에도, 대기시간 내에 다른 호스트로부터 클러스터 헤드 선언 메시지를 수신하는 호스트는 더 이상 클러스터 헤드 선출과정에 참여하지 않으므로 2홉 클러스터의 정의를 만족한다. 일반적으로 호스트들의 이동에 의해 클러스터 헤드간 인접하는 경우와 클러스터 헤드를 유실하는 경우는 네트워크 전체에 분산되어 발생가능하므로, 제안하는 방법은 완전히 분산된 형태로 동작한다.

4. 실험 결과

본 논문에서 사용된 시뮬레이터는 Mesquite사에서 개발한 CSIM 18 엔진을 이용하여 개발되었다. CSIM 18은 프로세스 기반의 이산사건을 시뮬레이션하기 위한 라이브러리 루틴만을 가지고 있으므로, 시뮬레이션에 필요한 세부 사항들은 모두 C언어로 구현되었다. 본 시뮬레이터는 일정영역내에서 랜덤하게 이동하며 주기적으로 패킷을 교환하는 호스트들을 프로세스로 모델링하였으며 패킷교환을 통한 클러스터 재구성 과정을 사건중심기법(Event driven method)에 의해 수행하였다.

시뮬레이션은 300×300 및 500×500 그리드상에서 45개의 호스트를 생성하여 600초 동안 구현되었다. 모든 호스트들은 초기에 랜덤하게 배치되며, 랜덤한 방향으로 이동한다고 가정하였다. 시뮬레이션에서 호스트의 이동속도는 세가지 클래스(A,B,C)중 하나에 속하게 된다. 시뮬레이션을 위한 입력 파라미터들은 표 1과 같이 설정되었다.

표 1 시뮬레이션을 위한 입력 파라미터

입력 파라미터	값
호스트의 수	45
전송범위	10~230meter
이동속도	A(0~5m/s),B(5~10m/s),C(10~15m/s)
Hello 전송 주기	3초

현재까지 클러스터 구조의 유지를 위해 제안된 방법들은 크게 두 종류로 나뉜다. 하나는 3계층과 4계층사이에 존재하며 제어메시지를 교환하는 방법들[1,6-8]이고, 다른 하나는 2계층에서 송신하는 프레임에 클러스터 상태정보를 삽입하여 처리하는 방법들[9,10]이다. 후자는 전자에 비해 특성상 클러스터 수가 많아지게 되며 클러스터 구조가 갖는 이점을 약화시키기 때문에 비교대상에서 제외하였다. 전자의 방법중에서 참고문헌 [6]에서 제안된 방법은 각 이동호스트 단위로 클러스터가 변화되는 것을 줄이기 위한 방법이고, 본 논문에서 제안하는 방법은 클러스터의 유지에 따르는 제어메시지의 감소를 목적으로 하므로 비교의 대상이 되지 않는다. 또한 참고문헌 [7]에서 제안된 방법은 LCC와 제어메시지수가 동일하고, 참고문헌 [8]의 방법은 LCC를 4홉 클러스터로 확장하기 위한 방법이므로 비교대상에서 제외하였다.

이러한 사항들을 고려하여, 본 논문에서는 클러스터를 유지하기 위해 전송되는 제어메시지수를 LCC[1]와 비교하였다. 이때의 제어메시지들은 그 용도에 따라서 방송되는 메시지와 unicast전송되는 메시지로 구분되지만 편의상 구분하지 않았다. 실험은 두 방법에서 전송하는 제어메시지의 수를 전송범위와 호스트의 수를 변화시켜

비교하였다.

그림 3에서는 본 논문에서 전송범위의 증가에 따라 제안하는 방법과 LCC에서 전송되는 평균 제어메시지 수를 보여준다. 제안하는 방법은 모든 전송범위에서 LCC보다 작은 제어메시지를 통해 클러스터구조를 유지한다. 이때 제어메시지는 LCC에 비해 약 50%정도 감소한다. LCC에서는 클러스터 구조를 유지하기 위해 클러스터 헤드와 멤버호스트들간에 양방향 메시지교환이 발생하므로, 클러스터 수와 클러스터 내의 멤버수에 따라 전송메시지수가 크게 좌우된다. 각 호스트들의 전송범위가 극히 작은 경우(=10meter)에는 호스트들간의 연결성이 전체적으로 작아지므로 많은 클러스터 헤드가 발생한다. 전송범위가 점차로 증가(=20meter)함에 따라 클러스터의 수는 약간 줄어들지만, 각 클러스터내에 포함된 호스트의 수가 증가하므로 LCC는 제어메시지의 수를 증가시킨다. 그러나 전송범위가 더 커지게 되면 (>30meter) 클러스터의 수가 급격히 감소하므로, 제어메시지의 수도 크게 감소된다.

반면에 제안하는 방법에서는 클러스터를 유지하기 위해 클러스터 헤드와 단일 클러스터에 소속된 호스트들만 단방향으로 메시지를 전송한다. 이때 클러스터 헤드는 제어메시지를 방송하고, 단일 클러스터에 소속된 호스트들은 unicast 전송한다. 전송범위가 작을때는 호스트들간의 연결성이 작고 클러스터 수가 많아지므로, 제어메시지의 수는 클러스터 수에 의해 영향을 받는다. 그러나 전송범위가 계속 증가하면 클러스터의 수가 크게 감소하므로, 제어메시지의 수는 클러스터의 수보다는 단일 클러스터에 소속된 호스트의 수에 의해 영향을 받는다. 단일 클러스터에 소속된 클러스터의 수는 전송범위의 증가에 따라 증가하지만, 호스트들의 위상에 따라 진동현상이 발생한다.

그림 4에서는 제어메시지를 방송메시지와 unicast메시지로 구분하여 실험한 결과를 보여준다. LCC와 제안된 방법 모두 호스트의 전송범위가 증가할수록 전송되는 boradcast메시지의 수는 감소한다. 이는 전송범위가 증가할수록 클러스터의 수가 감소하기 때문이다. 제안한 방법에서는 클러스터 헤드들만 메시지를 방송하므로 LCC에 비해 방송메시지의 수가 크게 감소된다. 반면에 LCC는 전송범위가 증가함에 따라 unicast메시지의 수가 크게 감소한다. 이는 전송범위가 증가함에 따라 클러스터의 수도 크게 감소하고 대부분의 보통 호스트가 클러스터들에 포함되기 때문이다. 그러나 제안하는 방법은 높은 전송범위에서 보다 많은 호스트들이 클러스터에 포함되므로 이들에 의한 unicast메시지수가 증가한다. 이때 발생하는 진동현상은 호스트들의 위상에 의한 것이다.

그림 5는 시뮬레이션 영역을 500meter×500meter로 확장하여 실험한 결과를 보여준다. 제안하는 방법은 모든 전송범위에서 작은 제어메시지를 유발하며, 50% 정도의 감소효과를 나타낸다.

그림 6은 전송범위를 각각 40meter와 80meter로 고정시킨 상태에서 호스트의 수를 변화시켜 제어메시지의 수를 측정한 결과를 보여준다.

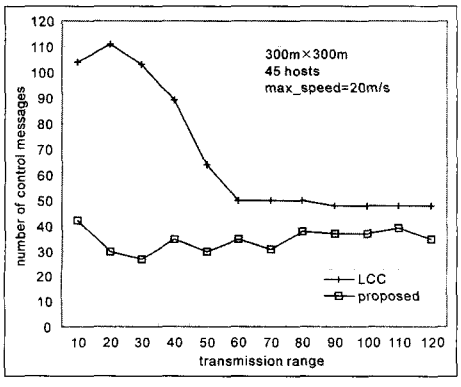


그림 3 전송범위에 따른 제어메시지 수(300m×300m)

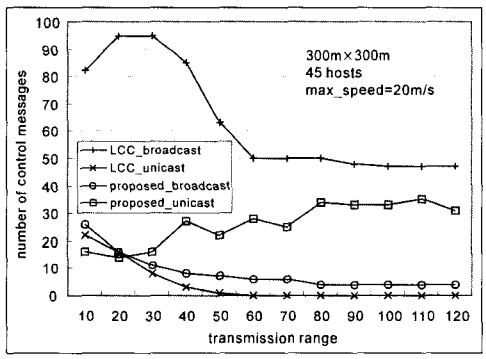


그림 4 전송범위에 따른 unicast 및 broadcast 메시지 수 (300m×300m)

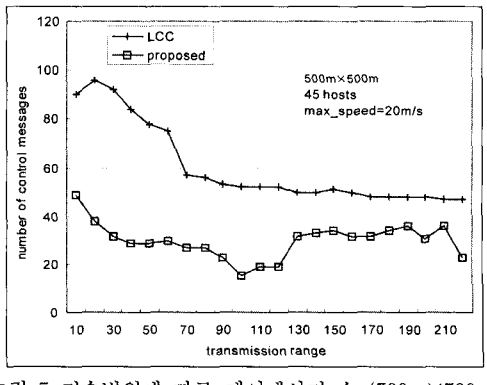


그림 5 전송범위에 따른 제어메시지 수 (500m×500m)

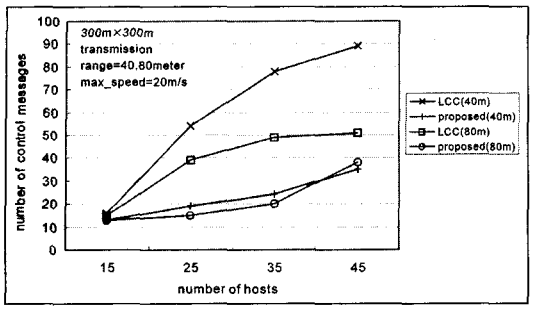


그림 6 호스트수에 따른 제어메시지수

일정한 전송범위내에서 호스트의 수가 증가하면 클러스터내의 호스트수가 증가하게된다. 따라서 LCC는 호스트수의 증가에 따라 제어메시지의 수도 크게 증가한다. 그러나 클러스터내의 호스트 수가 증가한다고 해도, 단일 클러스터에만 소속되는 호스트의 수가 증가하지는 않는다. 이는 호스트들이 랜덤하게 분포되기 때문이다. 그러므로 제안하는 방법은 호스트 수의 증가에 대해 제어메시지의 수가 크게 증가하지 않는다. 특히 전송범위 40meter에서 제안하는 방법과 LCC의 제어메시지수의 차이는 호스트수의 증가에 따라 계속 증가한다. 이를 통해 제안하는 방법이 좀더 확장성이 높다는 것을 알 수 있다.

5. 결 론

일반적으로 Ad hoc 네트워크에서 클러스터 구조를 이용하면 flooding되는 메시지의 재전송 횟수를 감소시킬수 있다. 그러나 기존의 클러스터 유지 방법들은 클러스터 구조를 유지하기 위해 많은 제어메시지 교환을 요구하므로 이러한 잇점을 크게 약화시킨다. 따라서 본 논문에서는 클러스터 구조의 잇점을 최대한 살리면서, 클러스터 정보의 획득 및 클러스터 재구성으로 인한 제어메시지 교환부하를 감소시키는 클러스터 유지 방법을 제안하였다. 시뮬레이션에서 제안하는 방법은 클러스터 구조를 유지하는데 있어 기존의 LCC방법에 비해 약 50% 정도의 제어메시지를 감소시켰다. 또한 제안하는 방법은 LCC에 비해 확장성이 높다는 것을 증명하였다. 향후 연구로는 전송되는 메시지의 신속하고 신뢰성있는 도달성을 만족하는 클러스터 유지방법을 연구할 예정이다.

참 고 문 헌

[1] C. Chiang, H. Wu, W. Liu, M. Gerla, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel(CGSR)," Proc. of IEEE SICON'97, 1997, pp. 192~211.

[2] M. Gerla and C. Chiang, "Multicluster, Mobile, Multimedia Radio Network," ACM-Baltzer Journal of Wireless Networks, Vol. 1, No. 3, pp. 255~265, 1995.

[3] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing for Multihop Wireless Ad Hoc Networks," Ad Hoc Networking, ed. C. E. Perkins, pp.139~168, Addison-Wesley, 2000.

[4] C. E. Perkins and E. M. Royer, "Ad hoc On-Demand Distance Vector Routing," Proc. of IEEE WMCSA 99, 1999, pp. 90~100.

[5] M. Jiang, J. Li, Y. C. Tay, "Cluster Based Routing Protocol," IETF draft, Aug. 1999, Work in Progress.

[6] C. R. Lin, M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," IEEE JSAC, 15(3), pp. 1265~1275, 1997.

[7] P. Basu, N. Khan, T. D. C. Little, "A Mobility Based Metric for Clustering in Mobile Ad Hoc Networks," Proc. of IEEE ICDCS 2001 Workshop on Wireless Networks and Mobile Computing, 2001, pp. 413~418.

[8] G. Chen, F. Garcia, J. Solano and I. Stojmenovic, "Connectivity Based k-hop Clustering in Wireless Networks," CD Proc. of IEEE Hawaii Int. Conf. System Science, 2002, p. 188c.

[9] T. C. Hou and T. J. Tsai, "An Access-Based Clustering Protocol for Multihop Wireless Ad Hoc Networks," IEEE JSAC, 19(7), pp. 1201~1210, 2001.

[10] M. Gerla, T.J. Kwon, G. Pei, "On Demand Routing in Large Ad Hoc Wireless Networks with Passive Clustering," Proc. of IEEE WCNC 2000, Chicago, IL, 2000, pp. 100~105.

[11] 유준, 김종권, "무선 Ad Hoc 망에서 인접 노드 정보 획득을 위한 제어 부하의 감소방안," 한국 정보과학회 추계 학술대회, 2002년 10월.

[12] J. Yoo, H. Gil, C. Kim. "INK: Implicit Neighbor Knowledge Routing in Ad Hoc Networks," Proc. of IEEE VTC 2003-spring, 2003, pp. 1826~1830.

[13] K. Liu, Jiandong Li, K. Mukumoto, A. Fukuda, "Adative Control Protocol in Mobile Wireless Ad Hoc Networks," APCCAS 2000, 2000, pp. 13~17.

[14] P. Krishna, N. H. Vadiya, M. Chatterjee, D. K. Pradhan, "A Cluster-based Approach for Routing in Dynamic Networks," ACM SIGCOMM Computer Communication Review, 49, pp. 49~64, 1997.

[15] S. Basagni, "Distributed Clustering for Ad Hoc Networks," International Symposium on Parallel Architectures, Algorithms and Networks, 1999, pp. 310~315.

왕 기 철

1997년 광주대학교 전자계산학과 졸업(학사). 2000년 목포대학교 멀티미디어공학과 졸업(석사). 2001년~현재 전북대학교 컴퓨터 통계정보 학과 박사과정. 관심분야는 이동컴퓨팅, Ad hoc 네트워크, 무선네트워크 보안

방 상 원

1985년 전남대학교 계산통계학과(이학사) 1988년 전남대학교 계산통계학과(이학석사). 1995년 전남대학교 전산통계학과(이학박사). 1987년~현재 송원대학 컴퓨터정보과 교수. 관심분야는 Mobile Computing, 컴퓨터그래픽, 소프트웨어공학

조 기 환

1985년 전남대학교 계산통계학과 졸업(학사). 1987년 서울대학교 계산통계학과 졸업(석사). 1996년 영국 Newcastle 대학교 전산학과 졸업(박사). 1987년~1997년 한국전자통신연구원 선임연구원. 1997년~1999년 목포대학교 컴퓨터과학과 전임강사. 1999년~현재 전북대학교 전자정보공학부 부교수 관심분야는 이동컴퓨팅, 컴퓨터통신, 무선네트워크 보안, 센서네트워크, 분산처리시스템