韓國IT서비스學會誌
第9卷 第3號
2010年 9月

무선 네트워크를 위한 하향식 클러스터링 프로토콜의 구현*

윤필중** · 김상경*** · 김창화****

Implementation of a Top-down Clustering Protocol for Wireless Sensor Networks*

Phil-Jung Yun** · Sangkyung Kim*** · Changhwa Kim***

■ Abstract ■

Many researches have been performed to increase energy-efficiency in wireless sensor networks. One of primary research topics is about clustering protocols, which are adopted to configure sensor networks in the form of hierarchical structures by grouping sensor nodes into a cluster. However, legacy clustering protocols do not propose detailed methods from the perspective of implementation to determine a cluster's boundary and configure a cluster, and to communicate among clusters. Moreover, many of them involve assumptions inappropriate to apply those to a sensor field. In this paper, we have designed and implemented a new T-Clustering (Top-down Clustering) protocol, which takes into considerations a node's density, a distance between cluster heads, and remained energy of a node all together. Our proposal is a sink-node oriented top-down clustering protocol, and can form uniform clusters throughout the network. Further, it provides re-clustering functions according to the state of a network. In order to verify our protocol's feasibility, we have implemented and experimented T-Clustering protocol on Crossbow's MICAz nodes which are executed on TinyOS 2.0.2.

Keyword : Top-down Clustering, Uniform Clustering, Node Density, TinyOS, MICAz

논문투고일 : 2010년 07월 16일 논문수정완료일 : 2010년 09월 04일 논문게재확정일 : 2010년 09월 10일

* 본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(NIPA-2010-C1090-1021-0001).

** (주)아로마소프트 기술연구소 전임연구원

*** 강릉원주대학교 컴퓨터공학과, 교신저자

**** 강릉원주대학교 컴퓨터공학과

1. 서 론

센서 노드의 제한적인 에너지는 센서 네트워크의 생존기간을 결정하는 가장 중요한 요소 중의 하나이며, 에너지의 효율적인 이용을 위한 연구가 많은 연구자들에 의해 수행되어 왔다[1-3, 9, 10]. 일반적으로 센서 노드의 에너지는 패킷의 송·수신 시 가장 많이 소모되며[8], 이러한 통신 작업에서의 에너지 효율을 향상시키기 위한 연구가 활발하게 진행되고 있다. 센서 네트워크를 위한 클러스터링(clustering) 프로토콜은 센서 네트워크의 노드들을 계층적인 구조로 구성하여 동작되도록 함으로써 네트워크 전체적인 에너지의 효율을 제고하고 생존기간을 늘리기 위해 연구되고 있다[1, 4, 5-7, 9, 10].

클러스터링 프로토콜은 물리적으로 근접해 있는 노드들을 하나의 클러스터로 묶고 센서 노드들을 클러스터의 대표자인 클러스터 헤드(cluster head)와 클러스터에 참여하는 클러스터 멤버(cluster member)로 분리하여 센서 네트워크의 구조를 계층화한다. 클러스터 헤드가 클러스터 멤버의 데이터를 병합하는 방법으로 전송되는 데이터의 양을 감소시켜 센서 네트워크의 에너지 효율을 향상시키며 주기적인 재 클러스터링 등의 방법으로 각 센서 노드가 균등하게 에너지를 소모하게 하여 센서 네트워크의 생존기간을 증가시킨다[9]. 기 연구된 클러스터링 프로토콜들은 각 센서 노드에서의 에너지 균등 소비를 통한 네트워크 생존시간 연장을 위하여 여러 가지 방법을 제안하고 있다. [5]에서는 각 센서 노드에 자율적으로 네트워크 상태를 반영하는 가중치를 할당하고 가중치가 높은 노드를 클러스터 헤드로 선택하여 클러스터를 형성하는 분산된 방법을 제안하고 있다. 가중치는 이웃한 센서 노드들의 수 등이 될 수 있으며, 이 경우 클러스터 내 데이터 병합을 거쳐 싱크 노드로 전달되는 데이터 패킷의 개수를 줄임으로써 에너지 효율성을 제고할 수 있다. [4, 6, 9]는 센서 네트워크의 베이스 스테이션이 클러스터 형성을 총괄적으로 제어하는 방법을 채택하고 있다. 이러한 중앙집중화된 제어를 통해 센서 네트워크 내 클러스터의 균등한 형성이 가능하도록 하며 각 센서 노드의 에너지 소비를 균등하게 함으로써 센서 네트워크의 생존시간을 늘릴 수 있다.

본 논문에서는 형성되는 클러스터의 균등성 제공을 위하여 T-Clustering(Top-down Clustering) 프로토콜을 제안한다. T-Clustering 프로토콜은 싱크노드 중심의 하향식이고 계층적인 클러스터링 프로토콜이다. 완전한 중앙집중형인 기존의 프로토콜들[4, 6, 9]과는 달리 T-Clustering 프로토콜에서 싱크노드는 클러스터의 형성의 시작과 최상위 레벨의 클러스터 헤드 선정에만 관여한다. 클러스터 헤드가 선정된 이후 클러스터의 구성은 자율적으로 이루어진다. 싱크노드에 의해 선정된 클러스터 헤드는 하위 레벨의 클러스터링에 참여하여 클러스터 헤드를 선정한다. 이와 같은 과정은 센서 네트워크 내의 모든 센서 노드가 클러스터에 참여하게 될 때까지 반복된다. 각 단계에서의 클러스터 헤드 선정은 노드의 밀집도, 즉 이웃 노드들의 수, 클러스터 헤드 간 거리, 노드의 잔여 에너지를 복합적으로 고려하여 이루어진다. 한편, 많은 클러스터링 프로토콜들이 제안되었으나 클러스터 범위의 선정 및 구성, 클러스터 간 통신 방안 등이 구현 관점에서 명확하게 제시되지 않았으며, 일부는 실제 환경에 적용하기 적합하지 않은 가정을 포함하고 있다[9, 10]. 본 논문에서는 실제적인 센서 노드에의 구현 관점에서 T-Clustering 프로토콜을 연구하고 설계하였으며, 적용 가능성을 검증하기 위하여 Crossbow 사의 MICAz 노드[12]에 TinyOS 2.0.2[14]기반으로 구현한 프로토콜을 적용하여 실험을 수행하였다.

본 논문의 구성은 다음과 같다. 제 2장에서는 관련 연구를 기술하며. 제 3장에서는 T-Clustering 프로토콜에 대해 상세하게 소개한다. 제 4장에서는 TinyOS2.0.2를 기반으로 한 프로토콜 구현 내용을 기술한다. 제 5장에서는 상용 센서 노드 상에 구현된 프로토콜의 실험 측정 결과를 기술하며, 제 6장에서 결론을 맺는다.

2. 관련 연구

클러스터링 프로토콜은 분산 알고리즘과 중앙집중형 알고리즘으로 분류할 수 있다.

HEED[1]는 각 노드에서의 분산 처리 방법을 이용한 클러스터 헤드 선정 방안을 제안한다. 클러스터 헤드의 선정은 국부 데이터만을 이용하여 이루어져야 하며 클러스터링을 위해 전송되는 데이터양이 적어야 할 뿐만 아니라 일정 시간 내에 동작이 완료되어야 한다. HEED의 경우 클러스터의 크기에 관계없이 일정 시간 내에 알고리즘이 종료되며 이웃 노드의 위치를 고려하지 않아도 되는 장점이 있다. 국부 데이터를 이용한 분산 처리 방식의 클러스터 헤드 선정 방안은 LEACH-C[10]에서 보다 많은 오버헤드를 필요로 하지 않으며, 선정된 클러스터 헤드와 싱크 노드간의 통신은 클러스터 헤드간의 멀티 홉을 구성하여 이루어지므로 LEACH에 비해 높은 에너지 효율을 가진다. 하지만 클러스터 헤드 선정 방안을 제외하고는 LEACH의 방법을 따르며 HEED의 경우 클러스터가 네트워크에 균등하게 분포되기가 어렵다. 또한, 노드들의 에너지가 전체적으로 낮을 때 동시에 선정되는 클러스터 헤드의 수가 많아지므로 *AMRP(Average Minimum Reachability Power)*의 연산[4]에서 많은 에너지가 소모된다.

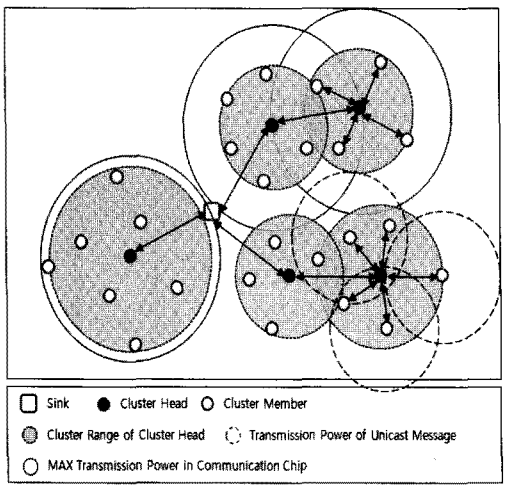
LEACH-C(LEACH-Centralized)[10]는 노드들이 자율적으로 클러스터를 구성하는 LEACH[9]에서 발생할 수 있는 비균등 클러스터 형성 문제를 해결하기 위해 제안된 중앙집중형 클러스터링 프로토콜이다. 비균등 클러스터 형성 문제는 클러스터에 따라 클러스터를 구성하는 멤버 노드들의 수 및 각 클러스터와 싱크 노드간 거리가 많이 다를 수 있음을 의미한다. 비균등 클러스터는 클러스터별로 소모되는 에너지의 균일성을 보장할 수 없다. 이러한 문제를 해결하기 위해 LEACH-C에서의 클러스터의 형성은 각 노드가 아닌 베이스 스테이션(싱크 노드)에서 이루어진다. LEACH-C는 클러스터 헤드의 선정 시 각 노드의 위치와 잔여 에너지의 정보를 이용하여 최적의 클러스터를 형성한다. 그러나 위치와 잔여 에너지 정보를 싱크 노드로 주기적으로 전송해야 하는 오버헤드가 발생하며 대부분 LEACH에서의 가정을 따르므로 LEACH에서와 같은 환경적 조건이 충족되어야 한다.

BCDCP(Base Station Controlled Dynamic Clustering Protocol)[4]은 베이스 스테이션이 클러스터 헤드 선정 및 클러스터 구성을 전적으로 담당하는 프로토콜이다. 이를 위하여 각 클러스터의 형성 단계 동안 베이스 스테이션은 네트워크 내 모든 노드들로부터 현재의 에너지 상태에 관한 정보를 받아야 한다. 베이스 스테이션은 이 정보에 기반하여 에너지 레벨이 평균 이상인 노드들의 집합을 만들고, 이 중에서 충분한 에너지를 가지는 노드들을 클러스터 헤드로 선정하게 된다. 먼저 네트워크를 2개의 클러스터로 나누고 미리 정해진 클러스터의 개수가 만들어질 때까지 클러스터 분리 작업을 반복한다. 매 분리 단계에서 클러스터 헤드간 거리가 최대가 되고, 클러스터 헤드의 부하가 균등하게 분산될 수 있도록 클러스터 헤드가 선택된다. BCDCP는 [7]에서 기술된 클러스터링 기술을 이용하여 균형 있는 클러스터 형성을 제공한다. 그러나 이 프로토콜은 베이스 스테이션이 클러스터 구성에 관한 모든 것을 수행하기 때문에 클러스터 형성 단계가 아닐 때 새롭게 노드가 추가되거나 노드의 장애가 발생된다면 적절하게 대응할 수가 없다. 또한, 클러스터 형성을 위해 매 단계마다 네트워크 내의 모든 노드들이 자신의 상태 정보를 포함하는 제어 패킷을 베이스 스테이션에 전달하여야 하므로 네트워크 규모가 커질 경우 확장성에 제약이 있을 수 있다.

3. T-Clustering 프로토콜

T-Clustering 프로토콜은 하향식 접근방법으로 클러스터를 형성하며, 클러스터 헤드들간의 멀티 홉 라우팅으로 싱크 노드와 센서 네트워크가 통신할 수 있도록 한다. T-Clustering 프로토콜에 의

해 형성되는 센서 네트워크의 구조는 [그림 1]과 같다. T-Clustering 프로토콜은 잔여 에너지가 많고, 클러스터 간의 중첩을 줄이기 위해 다른 클러스터 헤드와 먼 거리에 위치하며, 데이터 병합 연산의 효율을 위해 노드 밀집도가 높은 노드를 클러스터 헤드로 선정한다. 클러스터 헤드는 클러스터링 라운드 별로 갱신된다.



[그림 1] T-Clustering 프로토콜 네트워크 구조

T-Clustering 프로토콜에서는 각 노드에서의 에너지 효율을 높이고 센서 네트워크에서의 충돌을 감소시키기 위해 최적화된 전송 전력 수준으로 통신하는 방법[5]을 사용하며, 최적화된 전송 전력의 연산은 하위 계층에서 제공되는 것을 가정한다.

3.1 클러스터 헤드 선정

클러스터 헤드 선정 메커니즘에서는 *클러스터링 라운드(clustering-round)*라는 시스템 변수를 사용한다. *클러스터링 라운드*는 클러스터 형성 과정에서 클러스터링 절차를 확인하기 위한 식별자이며 클러스터 형성 시 싱크 노드에서 순차적으로 증가된다. 클러스터 헤드 선정은 클러스터 헤드 후보 선정 단계와 클러스터 헤드 선정 단계로 분리된다.

3.1.1 클러스터 헤드 후보 선정

클러스터 헤드 후보 선정 단계는 클러스터 헤드가 될 수 있는 후보 노드를 선정하고 클러스터 헤드 후보 노드들의 정보를 수집하는 단계이며, 싱크 노드나 싱크 노드로부터 클러스터 헤드 선정 권한을 부여 받은 클러스터 헤드에서 수행된다.

클러스터 헤드 후보 선정 작업을 시작하는 노드는 *CHC(Cluster Head Candidate)* 메시지를 방송하고 *CHCR(CHC Reply)* 메시지를 수신하기 위해 일정 시간 동안 대기한다. *CHC* 메시지에는 자신의 *클러스터링 라운드*가 삽입된다. *CHC* 메시지를 수신한 노드는 먼저 해당 메시지의 *클러스터링 라운드*와 자신의 *클러스터링 라운드*를 비교한다. 자신의 *클러스터링 라운드*가 수신된 *CHC* 메시지의 *클러스터링 라운드*보다 크거나 같다면, 해당 메시지를 폐기하고, 그렇지 않다면 자신의 상태 정보를 *CHCR* 메시지에 실어 응답한다. 상태 정보는 노드 밀집도, *CHC* 메시지 전송 노드와의 거리, 그리고 자신의 잔여 에너지로 구성된다. 노드 밀집도는 하위 계층에 의해 수집되는 것으로 가정한다. 거리 정보는 IEEE 802.15.4 표준[11, 13]에서 규정된 *RSSI(Received Signal Strength Indication)* 값을 이용하여 간단하게 연산할 수 있다. *CHCR* 메시지를 전송한 노드들은 클러스터 헤드 후보로서 관리된다.

3.1.2 클러스터 헤드 선정

클러스터 헤드 후보 선정 작업 이후 클러스터 헤드 선정 단계가 진행된다. 만일 클러스터 헤드 선정 단계를 진행할 때 클러스터 헤드 후보가 없다면, 클러스터 헤드 후보 선정 절차를 진행했던 클러스터 헤드는 자신의 상태를 클러스터 멤버로 변경한 후 싱크 노드로 그 사실을 알리며, 클러스터 참여 절차를 수행한다. 클러스터 헤드 후보가 존재할 때 클러스터 헤드 선정 작업이 계속 수행된다. 먼저 각 클러스터 헤드 후보로부터 수집된 정보를 기반으로 식 (1)을 이용하여 클러스터 헤드 후보에 대한 CH_{select} 값을 연산한다.

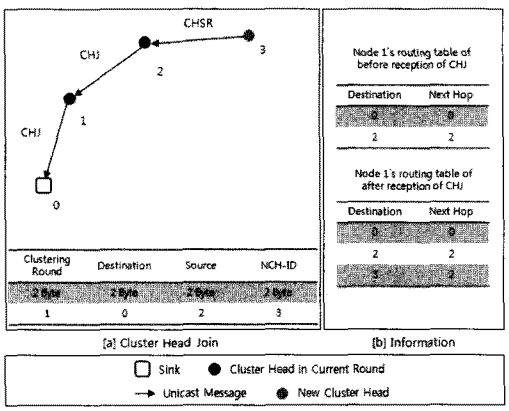
$$CH_{select} = W_{de}D_e + W_{di}D_i + W_eE \quad (1)$$

식 (1)에서 D_e는 밀집도, D_i는 현재 노드에서 해당 클러스터 헤드 후보까지의 거리, E는 해당 클러스터 헤드 후보의 잔여 에너지를 의미한다. 각 값은 각각의 최대값이 다르므로 연산 시에는 해당 값들의 편차값을 *0*에서 *1*사이의 값으로 정규화 하여 연산한다. W_{de}, W_{di}, W_e 값은 각각의 값에 대한 가중치 값을 의미하며 *0*에서 *1*사이의 값으로 정의되고 센서 네트워크가 배치되는 환경이나 에너지 효율 정책에 따라 결정된다. CH_{select}의 연산 결과는 CH_{select} 테이블로 구성되며 모든 연산이 종료된 후 내림차순으로 정렬된다. 클러스터 헤드 선정 절차는 CH_{select} 테이블을 순차적으로 탐색하여 CH_{select} 테이블에서 현재 선택된 노드로 *CHS (Cluster Head Select)* 메시지를 전송함으로써 시작된다.

CHS 메시지를 수신한 노드는 자신의 주변에 현재 클러스터링 라운드에서의 클러스터 헤드가 존재하는지의 여부를 확인하여 존재하면 *node-type*을 '*cluster-member*'로, 그렇지 않으면 '*cluster-head*'로 설정한 후 *CHSR(CHS Reply)* 메시지로 응답한다. *CHSR* 메시지 전송 노드는 자신의 클러스터링 라운드를 수신 CHS 메시지의 해당 값으로 갱신한다. *node-type*이 '*cluster-head*'인 경우 라우팅 테이블에 목적지를 싱크 노드로, 다음 홉(next hop)을 CHS 메시지 전송 노드로 하는 새로운 엔트리를 등록한다. 그리고 클러스터 헤드 테이블에 *CHS* 메시지를 전송한 노드의 ID를 삽입한다. 클러스터 헤드 테이블은 클러스터 복구 시 이상 상황이 발생한 노드와의 링크가 클러스터 헤드와의 링크인가를 판단하기 위해 사용된다.

싱크 노드가 *CHSR* 메시지를 수신하였고 *CHSR* 메시지의 *node-type*이 클러스터 헤드라면 헤드 선정 순서를 관리하기 위한 *FIFO queue*에 *CHSR* 메시지를 전송한 노드의 ID를 삽입한다. *FIFO queue*의 삽입 연산 후 라우팅 테이블에 목적지와 다음 홉을 *CHSR* 메시지를 전송한 노드로 삽입하고, 클러스터 헤드 테이블에 *CHSR* 메시지를 전송한 노드의 ID를 삽입한다. 모든 작업이 종료된 후 CH_{select} 테이블 탐색 알고리즘을 다시 수행한다.

클러스터 헤드가 *CHSR* 메시지를 수신하였고 *CHSR* 메시지의 *node-type*이 클러스터 헤드라면 다운-링크에 대한 라우팅 경로로 *CHSR* 메시지를 전송한 노드에 대한 경로를 삽입하고 *CHSR* 메시지를 전송한 노드의 ID를 *새로운 클러스터 헤드 ID(New Cluster Head ID : NCH-ID)*로 *CHJ(Cluster Head Join)* 메시지에 패킷화하여 싱크 노드로 전송한다. *CHJ* 메시지 전달 과정은 [그림 2]와 같다.



[그림 2] CHJ 메시지 전달 과정

CHJ 메시지가 싱크 노드로 전달되면서 멀티 홉 경로상의 클러스터 헤드들의 라우팅 테이블에 새로 선정 된 클러스터 헤드에 대한 경로가 추가된다. 클러스터 헤드 선정 권한을 부여 받은 클러스터 헤드, 즉 [그림 2]에서의 2번 클러스터 헤드는 *CHJ* 메시지를 전송한 후 CH_{select} 테이블 탐색 결과에 따라 헤드 선정 작업을 계속 진행한다. CH_{select} 테이블의 탐색이 모두 종료 되었다면 싱크 노드의 경우 *FIFO queue*의 헤드에 위치한 클러스터 헤드 ID로 *CHSS(Cluster Head Select Start)* 메시지를 전송하여 해당 클러스터 헤드에 클러스터 헤드 선정 권한을 부여한다. 클러스터 헤드의 경우 자신의 *클러스터링 라운드*를 *CJC(Cluster Join Com-*

mand) 메시지로 패킷화 하여 인접한 노드들로 방송한다. *CJC* 메시지를 수신한 노드들은 클러스터 참여 메커니즘을 실행한다.

클러스터 헤드 선정 권한을 부여 받은 클러스터 헤드는 *CJC* 메시지를 방송한 후 싱크 노드에 클러스터 헤드 선정 작업이 종료되었음을 알리고 선정 권한을 반납한다. 싱크 노드는 해당 클러스터 헤드 정보를 *FIFO queue*에서 삭제하고 *FIFO queue*의 위치한 다음 클러스터 헤드에 대해 클러스터 헤드 선정 권한을 부여한다. 만약 *FIFO queue*가 비어 있다면 현재 *클러스터링 라운*드에서의 클러스터 형성 작업이 모두 종료 되었다는 것을 의미한다. 싱크노드는 클러스터 헤드 선정 권한을 순차적으로 부여함으로써 클러스터의 영역이 중첩되는 상황을 방지한다.

3.2 클러스터 참여

클러스터 참여 메커니즘은 클러스터 멤버가 자신을 기준으로 가까운 거리에 존재하며 잔여 에너지 량이 많은 클러스터 헤드를 검색하고 해당 클러스터에 참여하는 과정을 포함한다. 클러스터 헤드 검색은 해당 노드가 클러스터 멤버가 되었을 때 처음 실행되는 단계이며 T_{SEARCH} 시간 동안 대기한 후 시작된다. T_{SEARCH}는 식 (2)와 같이 정의된다.

$$T_{SEARCH} \geq T_{SELECT} \times N_{QW} + T_{SMR} \qquad (2)$$

T_{SEARCH}는 클러스터 헤드 선정 권한을 받은 클러스터 헤드가 클러스터 헤드 선정을 마칠 수 있는 최대 시간을 의미하며 N_{QW}는 클러스터 헤드 선정 권한을 받은 클러스터 헤드가 클러스터 헤드 선정 권한을 반납하였을 때 싱크 노드의 *FIFO queue*에 대기하고 있는 노드의 수로서 경험적인 값을 이용한다. T_{SMR}의 경우 각 클러스터 멤버에서의 최대 랜덤 대기 시간을 의미한다.

클러스터 멤버는 클러스터 헤드 검색 단계에서 수집한 정보를 기반으로 식 (3)을 이용하여 CH_{search}를 연산한다.

$$CH_{search} = W_{di}(1 - D_i) + W_e E \qquad (3)$$

D_i는 현재 노드에서 해당 클러스터 헤드까지의 거리, 그리고 E는 해당 클러스터 헤드의 잔여 에너지를 의미하며 거리의 경우 가까울수록 높은 값을 가져야 하므로 $1-D_i$로 표현된다. W_{di}와 W_e는 각각의 값에 대한 가중치를 의미한다. 클러스터 헤드 선정에서와 같이 각각의 값은 *0*에서 *1*사이의 값으로 정규화 되며 가중치 또한 *0*에서 *1*사이의 값을 가진다. 연산된 CH_{search} 값은 CH_{search} 테이블로 구성되며 내림차순으로 정렬된다. 클러스터 멤버는 CH_{search} 테이블을 순차적으로 탐색하며 *CJ(Cluster Join)* 메시지를 전송하여 클러스터에 가입한다.

3.3 클러스터 복구

클러스터 헤드가 클러스터 헤드간의 멀티 홉 링크의 장애 사실을 확인하게 되면, 먼저 끊어진 링크에 위치하는 클러스터 헤드 ID를 클러스터 헤드 테이블에서 삭제한다. 그리고 해당 클러스터 헤드의 ID를 다음 홉으로 가지는 모든 경로 정보를 라우팅 테이블에서 삭제한다. 이후 링크 장애 사실을 싱크 노드를 향하여 통지한다. 링크 장애 사실을 통지 받은 경로 상의 클러스터 헤드들은 라우팅 테이블을 갱신한다. 싱크 노드는 *클러스터링 라운*드를 증가시킨 후 이상 상황을 발견한 클러스터 헤드에 클러스터 헤드 선정 권한을 부여하여 하위 클러스터를 지역적으로 복구할 수 있도록 한다.

4. 구현

T-Clustering 프로토콜은 TinyOS 2.0.2[14]기반으로 구현되었으며, [그림 3]은 T-Clustering 프로토콜 소프트웨어의 구조를 나타낸다.

ClusteringEngineC 컴포넌트는 *ClusteringControl* 인터페이스에 의해 클러스터 형성 기능을 제공한다. 초기의 클러스터 형성과 재 클러스터링이 *ClusteringControl* 인터페이스를 이용하여 이루어지며 *ClusteringControl* 인터페이스는 상위 계층에서의 요청으로 제어할 수 있으므로 각 라운드의 주기나 예측하지 못한 문제 또한 상위 계층에서의 필요성 등에 의해 언제든지 재 클러스터링이 가능하다.

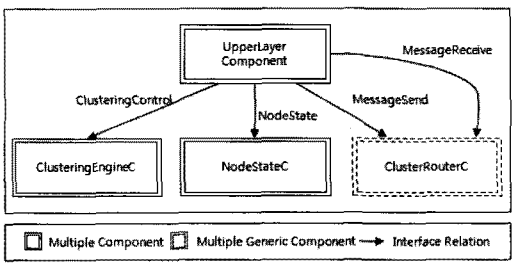
NodeStateC 컴포넌트는 현재 노드의 상태 정보를 저장하는 컴포넌트이며 *NodeState* 인터페이스에 의한 요청에 따라 상태 정보를 요청 측에 전달하거나 변경하는 기능을 제공한다. <표 1>은 *NodeState* 인터페이스를 나타낸다.

〈표 1〉 NodeState 인터페이스

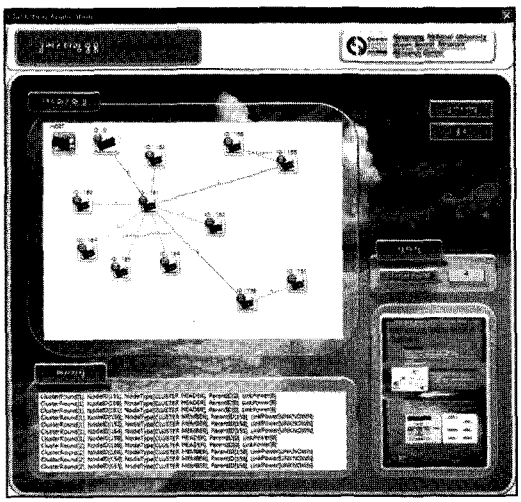
Command and Event	Function
Command uint8_t getState()	노드의 상태를 반환
Command uint8_t getClusteringRound()	노드의 클러스터링 라운드를 반환
Command error_t requestState(uint8_t reqState)	노드의 상태를 변경
Command error_t requestClusteringRound (uint8_t reqClusteringRound)	노드의 클러스터링 라운드를 변경

NodeState 인터페이스는 클러스터링 라운드나 노드의 상태를 다른 컴포넌트로 전달하거나 변경할 수 있도록 설계되었다. [그림 3]에서 사용되는 *NodeState* 인터페이스의 경우 상위 계층에 자신의 상태를 참조할 수 있도록 하며 클러스터 형성 과정에서 가장 많이 사용되는 인터페이스이다.

ClusterRouterC 컴포넌트는 *MessageSend*와 *MessageReceive* 인터페이스에 의해 상위 계층에서의 메시지 전송 요청을 처리하고 수신한 메시지를 상위 계층으로 전달하는 기능을 제공한다. 또한, 다른 노드로의 메시지에 대한 라우팅 기능을 제공한다. *ClusterRouterC* 컴포넌트의 경우 *generic* 컴포넌트로 설계되어 있다. TinyOS에서 *generic* 컴포넌트는 컴포넌트를 재사용할 수 있도록 하며, 중복되어 생성되는 *generic* 컴포넌트의 경우 객체지향 개발 방법론에서 클래스의 경우처럼 다른 컴포넌트의 성격을 가지게 된다. 일반적으로 상위 계층에서 사용하는 메시지의 종류가 여러 개이므로 *ClusterRouterC* 컴포넌트가 *generic* 컴포넌트로 설계된다. 즉, T-Clustering 프로토콜은 상위 계층에서 사용하고자 하는 메시지의 수만큼의 *ClusterRouterC* 컴포넌트를 제공하여 상위 계층에 다중화 기능을 제공한다. *ClusterRouterC* 컴포넌트는 다중화 기능을 제공하기 위해 생성 시 상위 계층으로부터 고유의 식별자를 부여받으며 이러한 식별자는 전송되는 메시지의 *type* 필드에 기입된다.



[그림 3] T-Clustering 프로토콜 소프트웨어 구조



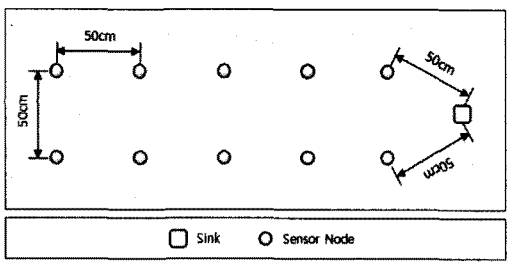
[그림 4] 응용 프로그램

T-Clustering 프로토콜의 응용 프로그램은 T-

Clustering을 PC 환경에서 테스트하기 위해 제작되었다. 응용 프로그램은 Microsoft Visual Studio 2005 에서의 Visual C++ MFC로 프로그래밍 되었으며, [그림 4]는 구현된 응용 프로그램을 보여준다. 응용 프로그램은 현재 네트워크에서의 클러스터링 라운드와 노드의 상태, 네트워크에서의 링크, 그리고 각 링크에서의 최적화된 전송 전력 수준을 나타낸다.

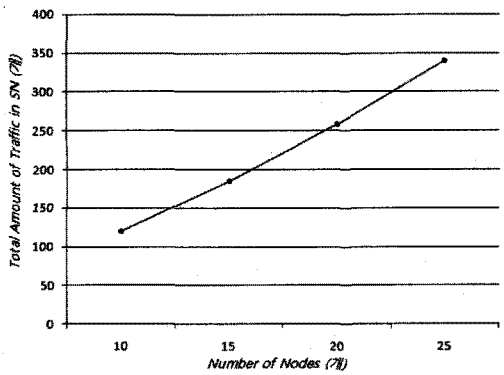
5. 실험

본 논문에서는 T-Clustering 프로토콜이 네트워크 내 각 노드의 에너지 소비 균등성을 제공하는 것을 확인하기 위해 Crossbow 사에서 개발된 MICAz 노드[12]에 구현된 T-Clustering 프로토콜 소프트웨어를 포팅하여 검증하였다. 이를 위하여 클러스터 형성 시 발생되는 노드 유형별, 그리고 네트워크 규모의 증가에 따른 제어 트래픽의 양을 측정하였으며, 클러스터링 라운드 별 노드의 잔여 에너지 편차를 측정하였다. 실험 환경은 [그림 5]와 같다. 각 노드는 실내에서 50cm의 간격으로 배치되었으며, 그 간격은 노드의 수나 측정 목표에 따라 일정하게 유지하였다.

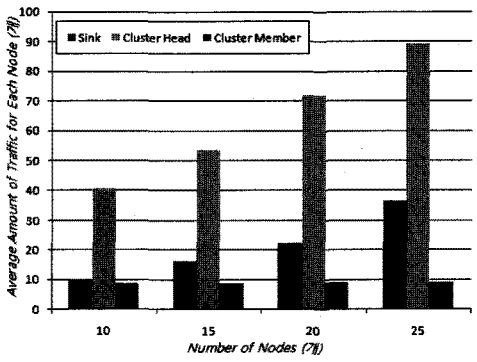


[그림 5] 측정 환경

클러스터 형성 시 발생되는 제어 패킷의 양은 싱크 노드를 포함하여 10, 15, 20, 25개의 센서 노드로 구성된 센서 네트워크에서 측정되었으며, 측정 결과는 [그림 6]과 같다. 실험 환경의 제약으로 인해 현재 모든 노드가 직접적으로 통신 가능한 위치에 배치되었으므로 1개의 클러스터 헤드가 선정되었다. 10개의 노드로 구성된 센서 네트워크에서 평균 *120.3*개의 제어 패킷이 발생하였고, 25개의 노드로 구성된 센서 네트워크에서는 평균 *339.6*개의 제어 패킷이 발생하였다. [그림 7]은 노드 타입별 측정 결과를 보여 준다. 클러스터 헤드에서 가장 많은 제어 트래픽이 발생하였으며, 네트워크 규모에 따라 트래픽이 증가하는 것을 확인할 수 있다. 반면, 클러스터 멤버에 대해서는 클러스터 헤드 후보 상태에서의 응답 작업과 클러스터 참여 작업을 수행하는데 해당 작업들은 이웃 노드들과의 상호작용에 의해서만 영향을 받아 네트워크 규모에 따른 트래픽의 증가가 없었다.



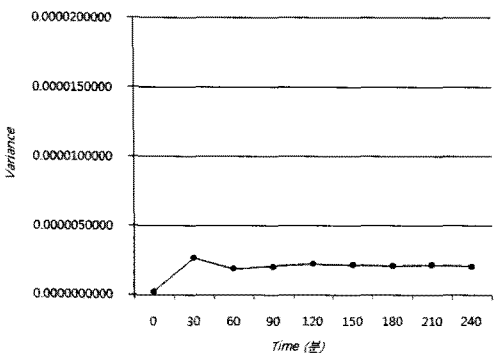
[그림 6] 클러스터 형성 시 발생되는 제어 트래픽



[그림 7] 노드 유형별 제어 트래픽

T-Clustering 프로토콜은 균등한 에너지 소모를 위해 클러스터링 라운드별로 새로운 클러스터 헤드를 선정하며, [그림 8]은 클러스터링 라운드별로

클러스터 헤드를 선정하였을 때의 잔여 에너지 편차를 보여준다. [그림 8]에서의 측정은 싱크 노드를 제외한 10개의 노드로 이루어졌으며, 편차는 클러스터링 라운드에서의 모든 노드에 대한 분산을 사용하였다. 분산의 연산은 실험된 데이터를 모집단으로 사용하여 이루어졌으며 클러스터링 라운드별 잔여 에너지를 측정하기 위해 클러스터링 라운드의 주기를 30분으로 설정하고 클러스터링 라운드의 시작점에서 잔여 전압을 측정하였다. 최대 *0.001V*의 차이로 각 노드를 배치하였으므로 시작점에서는 낮은 분산 값을 가진다. 클러스터링 라운드별 잔여 에너지 편차는 일정 지점에서 수렴하는 것을 확인할 수 있는데 이것은 T-Clustering 프로토콜이 에너지 소비 균등성을 보장한다는 것을 의미한다. [그림 8]에서 분산 값의 증가 원인은 크게 두 가지가 있으며, <표 2>는 그 원인을 분석하기 위한 시작지점에서 90분 지점까지의 측정 데이터와 선정된 클러스터 헤드를 보여준다.



[그림 8] 클러스터 라운드별 잔여 에너지 편차

<표 2>의 시작 지점에서 90분 지점까지 측정된 전압과 음영으로 표시된 부분은 해당 클러스터링 라운드에서 선정된 클러스터 헤드를 보여준다. 먼저 시작 지점을 보면 대체적으로 균일한 에너지를 가지는 노드 중에 8번 노드가 클러스터 헤드로 선정되는 것을 확인할 수 있는데 이것은 첫 번째 분산 값 증가의 원인이다. 즉, 분산 값이 작다는 것은 노드들의 잔여 에너지 량이 균일하다는 것을 의미하므로 이 경우 한 노드가 클러스터 헤드로 선정되어 많은 에너지를 소모하게 되면 분산 값이 증가하게 된다.

30분 지점에서는 가장 많은 잔여 에너지를 가지는 1번 노드가 클러스터 헤드로 선정되는 것을 확인할 수 있으며 이로 인하여 분산 값이 감소하는 것을 [그림 8]에서 확인할 수 있다. 60분 지점에서 선정되는 클러스터 헤드를 보면 잔여 에너지의 양이 가장 적은 8번 노드가 클러스터 헤드로 선정되는 것을 확인할 수 있으며 이것이 두 번째 분산 값 증가의 원인이다. 즉 현재 클러스터 헤드 선정 기준은 거리, 밀집도, 잔여 에너지이며 [그림 8]에서 각각의 가중치 값이 *1*로 설정되어 있으므로 잔여 에너지의 편차보다 다른 값의 편차가 커지게 되면 CH_{select} 값이 커지게 되므로 60분 지점에서 8번 노드가 클러스터 헤드로 선정된다. 90분 지점에서 잔여 에너지의 편차가 충분하게 발생하므로 잔여 에너지가 많은 2번 노드가 클러스터 헤드로 선정되는 것을 확인할 수 있다.

<표 2>의 분석 결과는 [그림 8]에서의 가중치 값의 변경을 통해 편차 값을 더 감소시킬 수 있다

〈표 2〉 클러스터링 라운드에서의 클러스터 헤드와 잔여 에너지(V)

Time(분) \ Node	1	2	3	4	5	6	7	8	9	10
0	3.252	3.252	3.251	3.251	3.251	3.251	3.252	3.252	3.251	3.251
30	3.182	3.181	3.179	3.178	3.178	3.18	3.179	3.176	3.178	3.178
60	3.145	3.145	3.143	3.141	3.143	3.144	3.143	3.141	3.142	3.142
90	3.11	3.11	3.108	3.106	3.107	3.108	3.107	3.106	3.106	3.107

는 것과 가중치 값의 설정이 중요하다는 것을 확인하게 해준다. 하지만 데이터 병합과 충돌 감소 또한 에너지 효율에 많은 영향을 미치므로 가중치 값은 해당 센서 네트워크 시스템이 요구하는 것이 충돌 감소로 인한 처리율인지 데이터 병합으로 인한 에너지 효율인지 에너지 소비의 균등성인지를 고려하여 결정되어야 한다.

실험을 통하여 T-Clustering 프로토콜이 클러스터간 부하균등성을 제공하여 네트워크 내 에너지 소비가 비교적 균일하다는 것을 확인할 수 있었다. <표 3>은 클러스터링 주체와 클러스터간 부하균등성 관점에서 기존 클러스터링 프로토콜들과 비교한 것이다.

6. 결 론

본 논문에서는 균등한 클러스터의 형성을 통해 에너지 효율성을 높이기 위한 싱크 노드 중심의 하향식 클러스터링 프로토콜을 제안하고 구현을 통해 실제 환경에의 적용 가능성을 실험하였다. T-Clustering 프로토콜은 중앙집중형과 분산형 클러스터링 프로토콜의 혼합형 프로토콜로서 노드 밀집도, 클러스터 헤드간 거리, 노드의 잔여 에너지 등을 복합적으로 고려하여 에너지 효율적으로 클러스터 헤드를 선정하고 클러스터를 구성할 수 있도록 하였다.

본 논문에서는 T-Clustering 프로토콜을 Tiny OS2.0.2 기반으로 구현하고, 상용 센서 노드에 포팅하여 실현 가능성을 검증하였다. 실험을 통해 T-Clustering 프로토콜은 클러스터 형성 과정에서 센서 네트워크를 구성하는 노드들 사이에 에너지 소비 균등성을 보장할 수 있다는 것을 확인하였다. T-Clustering 프로토콜은 센서 네트워크에서의 실제 환경을 고려하여 설계되고 구현되었으므로 실제 센서 네트워크 분야에서 에너지 효율적인 통신을 위한 네트워크 프로토콜로 사용할 수 있을 것으로 기대된다.

보유 센서 노드의 수와 실험 환경의 제약으로 인해 제한된 환경에서 실험을 실시하였으나 향후 실험 네트워크의 규모를 늘리고, 다양한 환경에서의 실험을 통하여 본 논문에서 제안한 가중치들의 적정한 값들을 도출하고 클러스터의 균등성과 에너지 효율성을 평가할 계획이다. 또한, 구현상의 제약으로 인해 기존의 프로토콜들과의 성능 비교 실험은 실시하지 못하였으나 추후 시뮬레이션을 통하여 수행할 예정이다.

참 고 문 헌

[1] Younis, O. and S. Fahmy, "HEED : A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad-hoc Sensor Networks",

〈표 3〉 기존 클러스터링 프로토콜들과의 정성적 비교

	LEACH[9]	LEACH-C[10]	HEED[1]	BCDCP[4]	T-Clustering
클러스터링 주체	분산형, 노드들이 클러스터링 라운드별 자율적으로 클러스터 형성	집중형, 노드들로부터 주기적으로 전송되는 정보를 이용하여 싱크노드가 모든 클러스터 헤드 선정	분산형, 국부적인 데이터만을 이용하여 각 노드들이 자율적으로 클러스터 형성	집중형, 클러스터 헤드와 멤버 노드들을 모두 싱크노드가 선정하여 구성	혼합형, 싱크노드는 최상위 레벨의 클러스터 헤드 선정에만 관여하며, 하위 레벨의 클러스터 헤드는 상위 레벨의 클러스터 헤드에 의해 선정됨
클러스터간 부하균등성	제공하지 못함	싱크노드가 각 노드의 위치 정보를 이용하여 클러스터 헤드를 선정하므로 균등성 제공	국부적인 정보만을 이용하여 클러스터를 형성하므로 균등성 제공이 어려움	싱크노드가 중앙집중형으로 네트워크 내 모든 클러스터를 구성하므로 균등성 제공 가능	균등성 제공

Proceedings of IEEE Transactions on Mobile Computing, (2004), pp.366-379.

[2] Yun, Phil-Jung, Changwha Kim, and Sang-kyung Kim, "Implementation and Perfromance Evaluation of ELM-MAC Protocol for Energy Efficiency in Sensor Networks", *Journal of the Korea Society for Simulation*, Vol.17, No.4(2008), pp.81-88.

[3] Shah, R. C. and J. M. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks", *Proceedings of IEEE Wireless Communication and Networking Conference*, (2002), pp.17-21.

[4] Krishnan, R. and D. Starobinski, "Efficient Clustering Algorithms for Self-Organizing Wireless Sensor Networks", *Ad Hoc Networks*, Vol.4, No.1(2006), pp.36-59.

[5] Basagni, S., "Distributed Clustering for Ad Hoc Networks", *Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms and Networks*, pp.310-315.

[6] Muruganathan, S. D., D. C. F. Ma, R. I. Bhasin, and A. O. Fapojuwo, "A Centralized Energy-Efficient Routing Protocol for Wireless Sensor Networks", *IEEE Radio Communications*, (2005), pp.S8-S13.

[7] Ghiasi, S., A. Srivastava, X. Yang, and M. Sarrafzadeh, "Optimal Energy Aware Clustering in Sensor Networks", *MDPI Sensors*, Vol.2, No.7(2002), pp.258-269.

[8] Madden, S. R., M. J. Franklin, J. M. Hellerstein, and Wei Hong, "TinyDB : An Acquisitional Query Processing System for Sensor Networks", *Proceedings of ACM Transactions on Database Systems*, (2005), pp. 122-173.

[9] Heinzelman, W. B., A. P. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, (2000), pp.10-19.

[10] Heinzelman, W. B., A. P. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks", *Proceedings of IEEE Transactions on Wireless Communications*, (2002), pp.660-670.

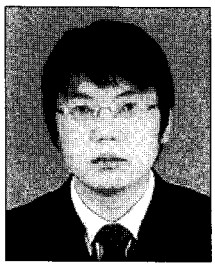
[11] Chipcon AF SmartRF®, "CC2420 Preliminary Datasheet(rev 1.2)", Available at *http://inst.eecs.berkeley.edu/~cs150/Documents/CC2420.pdf*

[12] Crossbow®, "MICAz Datasheet", Available at http://www.xbow.com/Products/Product_*pdf_files/Wireless_pdf/MICAZ_Datasheet.pdf*.

[13] IEEE Std 802 Part 15.4 : Wireless Medium Access Control (MAC) and Physical Layer (PHY), IEEE Std 802.15.4-2006, Sep. 2006.

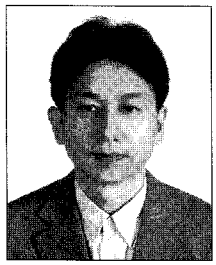
[14] TinyOS Open Technology Alliance, "What is TinyOS", Available at http://tinyos.net/*special/mission*.

◈ 저 자 소 개 ◈



윤 필 중 (philjung.yun@aromasoft.com)

강릉대학교 일반 대학원에서 컴퓨터 공학을 전공하고, 현재 (주) 아로마소프트 기술연구소에서 근무 중이다. 주요 관심분야는 해양 센서 네트워크, 데이터베이스, 안드로이드 플랫폼 등이다.



김 상 경 (skkim98@gwnu.ac.kr)

고려대학교에서 전자공학석사를 하고 동 대학에서 네트워크 전공으로 공학박사를 받았다. 1989년부터 2004년까지 주식회사 케이티에서 선임연구원(부장)으로 근무하였다. 1994년부터 1995년까지 미국 Bellcore에서 TINA 국제콘소시움 Core Team Member로 연구하였다. 2004년부터 현재까지 강릉원주대학교 컴퓨터공학과 부교수로 재직 중이며, 주요 연구관심 분야는 컴퓨터네트워크, 수중센서네트워크, 무선모바일네트워크 등이다.

김 창 화 (kch@gwnu.ac.kr)

고려대학교에서 전산학으로 이학석사학위를 받고 동 대학교에서 지식표현을 위한 모델과 방법을 제안하여 이학박사학위를 받았으며, 토론토 대학EIL 연구소와 Texas A&M 대학에서 각각 Post-Doc. 활동과 Visiting Scholar 활동을 한 바 있다. 현재 강릉원주대학교 컴퓨터공학과 교수로 재직 중이며 현재 강릉원주대학교 해양센서네트워크시스템기술연구센터 소장을 맡고 있다. 주요 연구관심분야로는 수중센서네트워크, 분산시스템, 온톨로지 등이며 특히 수중센서네트워크 분야를 집중적으로 연구하여 국내외 저널과 컨퍼런스에 논문 발표, 특허 출원, 소프트웨어 등록, 시제품 제작, 기술이전 등의 적극적인 산학활동에 임하고 있다.