공학석사 학위논문

*WSN*에서 위치좌표를 이용한 에너지 효율적인 라우팅 프로토콜에 관한 연구

A Study on Energy-Efficient Routing Protocol using Position Coordinate for WSN

지도교수 임 재 홍

*2007*년 *2*월

한국해양대학교 대학원

전자통신공학과

최 홍 석

本 論文을 崔洪碩의 工學碩士
學位論文으로 認准함.

委員長 : 梁 圭 植 (印)

委 員 : 沈 俊 煥 (印)

委 員 : 林 宰 弘 (印)

*2007*年 *2*月

韓國海洋大學校 大學院

電子通信工學科

목 차

Abstract

제 *1* 장 서 론 ····· 1

1.1 연구의 배경 ····· 1

1.2 연구의 목적 및 내용 ····· 4

제 *2* 장 위치좌표를 이용한 기존의 라우팅 프로토콜 ····· 6

2.1 Directed Diffusion ····· 6

2.2 GEAR ····· 10

제 *3* 장 제안된 라우팅 프로토콜 ····· 17

3.1 제안된 프로토콜의 개요 ····· 17

3.2 제안된 프로토콜의 동작 ····· 20

제 *4* 장 시뮬레이션 및 결과 분석 ····· 32

4.1 방향성 정의 알고리즘의 동작 검증 ····· 32

4.2 제안된 프로토콜과 GEAR의 성능 비교 ····· 34

제 *5* 장 결 론 ····· 39

참 고 문 헌 ····· 40

표 목 차

<표 3-1> 맥스포(주) TIP 50CM의 스펙 ····· 19
<표 3-2> 변수에 따른 방향성 설정 ····· 24
<표 3-3> 헬로 패킷의 포맷 ····· 26
<표 3-4> 상·하·좌·우로 방향성 설정 조건 ····· 30
<표 3-5> 노드가 이동하였을 때 발생하는 헬로 패킷의 포맷 ····· 31
<표 4-1> 시나리오 설정 ····· 36

그 림 목 차

<그림 1-1> WSN의 구조 ····· 1
<그림 1-2> 센서리아 WINS NG 2.0의 블록 다이어그램 ····· 4
<그림 2-1> Directed Diffusion의 단순화된 구조 ····· 7
<그림 2-2> interest와 event 메시지의 예 ····· 8
<그림 2-3> 구멍을 우회하기 위한 경로 학습 ····· 14
<그림 3-1> 방향성 정의 알고리즘의 개념 ····· 22
<그림 3-2> 예외상황의 처리 ····· 22
<그림 3-3> C++로 나타낸 방향성 정의 알고리즘 ····· 23
<그림 3-4> 각 노드에서 방향성이 설정된 모습 ····· 25
<그림 3-5> C++로 작성한 헬로 패킷 ····· 27
<그림 3-6> 변수 up이 활성화되었을 때 방향성 설정 ····· 29
<그림 4-1> 방향성 정의 알고리즘의 동작 검증 ····· 32
<그림 4-2> 각 노드별 송·수신 메시지 양의 비율 ····· 33
<그림 4-3> Directed Diffusion과 GEAR의 비교 ····· 35
<그림 4-4> 시뮬레이션 과정 ····· 37
<그림 4-5> 시뮬레이션 결과 그래프 ····· 38

Abstract

WSN(Wireless Sensor Network), which is fundamental technology for ubiquitous computing, is more restricted in resource such as battery than mobile ad-hoc network and most of working area can't be approached by the operators. Thus, efficient consumption of energy at each node is the most important field of research for WSN.

Previous routing protocols for WSN haven't completed to accommodate its features. Especially, big profit in energy consumption, global ID problem, etc. are expected when position coordinate is used, but there are rare protocols using it actively. In case of GEAR(Geographical Energy-Aware Routing), which represent protocol using position coordinate, don't expose feature of WSN because it was designed only for forwarding query messages and supposed fixed network environment.

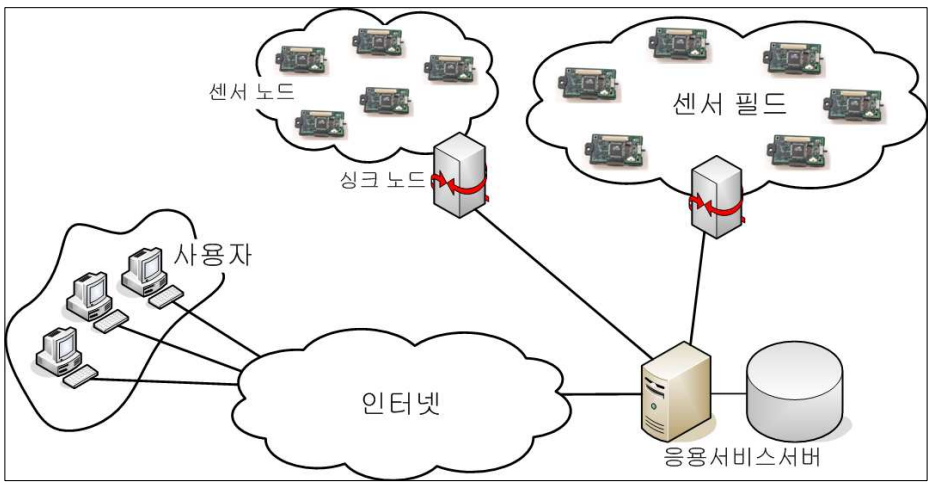
Proposed routing protocol in this paper defines direction of data, which route packet from each nodes to sink, based on position coordinate of both each node and sink. It hasn't global ID because of location-aware and data-centric routing, and prevent unnecessary consumption of energy by forwarding packet along direction.

제 1 장 서 론

1.1 연구의 배경

지난 2006년 2월, 정보통신부는 기존의 IT839 정책을 수정한 u-IT839 정책을 발표했다. 이는 8대 신규 서비스, 3대 인프라, 9대 신 성장 동력 간의 유기적인 연계를 통해 IT 산업의 경쟁력을 지속적으로 강화시켜 글로벌 IT 국가의 리더로 도약하려는 정부의 목표를 반영한 것이다[1].

WSN(Wireless Sensor Network)은 u-IT839 정책의 3대 인프라 중 하나인 USN(Ubiquitous Sensor Network)의 기반 기술로서 <그림 1-1>과 같이 일정 범위의 센서 필드를 구성하는 다수의 센서 노드, 센서 노드에서 감지된 환경데이터를 통합하는 싱크 노드, 통합된 데이터를 사용자에게 제공하는 응용서비스 서버로 구성된다[2].



<그림 1-1> WSN의 구조

<Fig. 1-1> Schematic for WSN

응용분야의 특성에 따라 수십 개에서 수천 개에 이르는 대량의 노드가 사람이 접근하기 어려운 지역에 감시·관찰 임무를 위해 배포되고, 센서를 통해 감지된 환경데이터는 네트워크 내부에서 보다 상위의 이벤트로 변환되어 원격의 관리자에게 전달된다. WSN의 주요 응용분야는 의료서비스, 전장에서의 운용, 위기감지, 재난방지, 환경감시, 토지감시, 로봇공학, 빌딩제어 등으로 매우 다양하며 그 특징은 다음과 같다[3].

- 고밀도 (High Density)

  WSN을 운용하는 주요 목적은 사람이 접근하기 어려운 지역을 감시·관찰하기 위함이다. 제한된 지역에서 보다 정확한 데이터를 얻기 위해 매우 많은 수의 노드를 무작위로 뿌린다. 따라서 센서 노드는 조밀하게 배포된다.

- 저비용 (Low Cost)

  센서 노드는 대량으로 배포되기 때문에 센서 노드로 사용되는 임베디드 보드는 비용이 저렴해야 한다.

- 제한된 자원 (Constrained Resources)

  WSN에서 가장 많은 제약을 받는 부분은 메모리와 배터리이다. 센서 노드로 사용되는 임베디드 보드는 그 물리적인 크기가 매우 작기 때문에 온보드(on-board) 타입의 메모리와 보드에 부착되는 배터리의 물리적 크기 역시 작을 수밖에 없으며 그 용량 또한 상당한 제약을 받는다[4].

- 자가-구성 (Self-organization)

  센서 노드는 무작위로 뿌려지고 예기치 못한 노드의 이동이 발

생할 수 있기 때문에 특정 토폴로지를 가정할 수 없다. 따라서 이웃 노드 간의 무선 통신을 통해 스스로 Ad-hoc 망을 구축할 수 있어야 한다.

- 고장 감내 (Fault Tolerance)

  임의의 특정 노드가 동작하지 않거나 노드의 이동으로 인해 망에 구멍(hole)이 발생하더라도 네트워크를 계속 유지할 수 있어야 하며 전체적인 네트워크의 성능에 영향을 주지 않아야 한다.

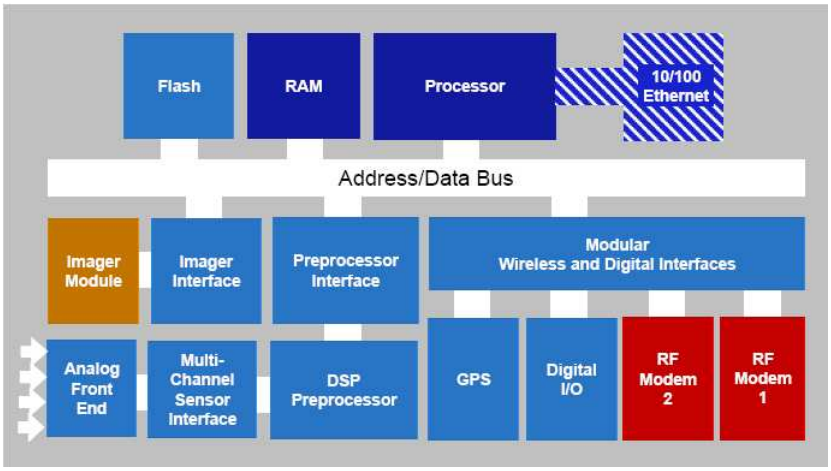
- 이동성 (Mobility)

  WSN은 특정지역을 감시·관찰하기 위해 운용된다. 사람의 손이 닿지 않는 자연환경에서 강우·강풍·지진 등의 자연재해나 야생 짐승에 의해 예기치 못한 노드의 이동이 발생할 수 있다.

위의 특징에서 보았듯이 WSN은 네트워크의 자가 구성을 위해 MANET(Mobile Ad-hoc NETwork) 기술을 요구한다. 그러나 기존 MANET 기술을 WSN에 그대로 적용시킬 수는 없다. MANET에서 노드의 메모리는 범용 데스크탑에 비해 제한적이긴 하지만 WSN에 비하면 매우 풍족한 편이다. 마찬가지로 MANET에서 노드의 배터리는 필요에 따라 교체·충전이 가능하지만 WSN의 배터리는 그렇지 못하다. 왜냐하면 센서 노드는 그 수가 수십에서 수천에 이를 정도로 매우 많으며 사람이 접근하기 어려운, 위험한 지역에서 동작하기 때문이다. 즉, WSN의 노드는 일회용 소모품의 성격을 지닌다. 따라서 센서 노드에서 자원 관리는 매우 중요한 연구 대상으로서 반드시 극복되어야 하는 부분이다[5].

다른 한편으로, MEMS(Micro-ElectroMechanical System) 기술이 발달하면서 GPS(Global Positioning System) 모듈이 센서 노드로

사용되는 임베디드 보드에 수용될 만큼 소형화되었고 미국의 센서리아(Sensoria)사와 UCLA(University of California, Los Ansgeles) 등에서 <그림 1-2>와 같이 GPS 모듈을 장착한 센서 노드를 개발하였다. 이에 따라 GPS 정보 중 위치좌표를 라우팅에 도입하는 새로운 패러다임이 등장하게 되었다[6][7][8].



<그림 1-2> 센서리아 WINS NG 2.0의 블록 다이어그램
<Fig. 1-2> Block Diagram of Sensoria WINS NG 2.0

1.2 연구의 목적 및 내용

본 연구의 목적은 각 노드의 위치좌표를 도입한 라우팅 프로토콜을 이용하여 각 노드에서 발생하는 통신 횟수를 줄임으로써 네트워크에서 소비되는 에너지의 양을 경감시켜 네트워크의 수명을 늘리는 것이다. 이를 위해 본 논문에서는 방향성 정의 알고리즘을 정의하고 알고리즘의 동작을 평가한다. 그리고 간단한 라우팅 프로토콜을 제안하고 여기에 방향성 정의 알고리즘을 적용하여 시뮬레이션하고 그 결과를 GEAR 프로토콜과 비교분석한다.

논문의 구성은 다음과 같다. 2장에서는 WSN을 위해 제안된 라우팅 프로토콜 중 위치좌표를 이용하는 대표적인 프로토콜인 Directed Diffusion과 GEAR에 대해 자세히 기술하고 평가한다. 3장에서는 본 논문에서 제안하는 라우팅 프로토콜에 대해 기술한다. 4장에서는 방향성 정의 알고리즘과 제안하는 프로토콜을 평가하고, 5장에서 결론을 맺는다.

제 2 장 위치좌표를 이용한 기존의 라우팅 프로토콜

2.1 Directed Diffusion

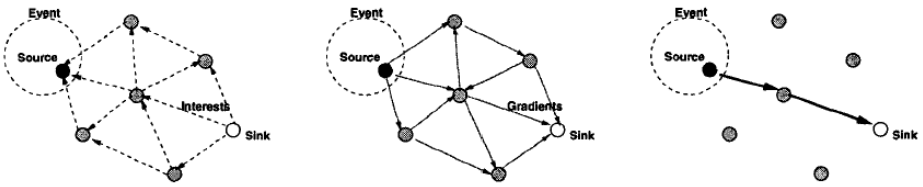
2.1.1 Directed Diffusion의 개요

Directed Diffusion은 신뢰성(robustness), 확장성(scaling), 에너지 효율성(energy-efficiency) 등의 요구를 충족시키기 위해 제안된 WSN의 대표적인 요구기반 라우팅 프로토콜이다. 관리자가 “지역 X에서 관찰되는 보행자는 몇 명인가?” 혹은 “지역 Y에서 차량은 어느 방향으로 이동 중인가?”와 같은 질문을 질의메시지의 형태로 네트워크에 전파하면 이 질문에 해당하는 지역의 노드들이 센서에서 감지되는 데이터를 질문에 답할 수 있는 메시지 형태로 변환하여 질의메시지의 경로를 그대로 거슬러 관리자에게 전달한다. 이때, 중간 노드들이 비슷한 내용의 데이터를 병합하는 역할을 한다. Directed Diffusion에서는 질의메시지를 interest, 응답메시지를 data, 응답메시지가 거슬러 오르는 경로를 gradient라고 정의한다[9].

Directed Diffusion의 중요한 특징은 interest 전달, 데이터 전달, 데이터 병합 등이 인접하는 노드들의 지역적 상호작용(localized interaction)에 의해서 이루어진다는 것이다. Directed Diffusion은 각 노드간의 상호작용을 통해 전역 ID가 아닌 데이터의 속성으로 각 노드를 식별하고 라우팅을 수행한다.

2.1.2 Directed Diffusion의 동작

Directed Diffusion은 앞서 언급했던 interest, data, gradient 외에 reinforcement를 포함하여 구성된다. interest와 data는 네이밍(naming) 규칙에 의해 속성-값 쌍으로 표현되며, interest는 사용자가 원하는 특정 데이터에 대한 설명을 가지고 네트워크로 전파된다. 각 노드에서는 interest에 부합하는 data가 있을 경우 "draw" 이벤트를 발생시키고, 이벤트를 발생시킨 노드는 interest를 전달한 노드로 gradient를 생성한다. 따라서 gradient는 사용자가 원하는 특정 데이터를 가진 노드에 대한 방향을 나타낼 수 있다. 감지된 데이터는 여러 개의 gradient 경로를 따라 interest의 발생 노드로 거슬러 오른다. 마지막으로, 각 노드는 네트워크 내부의 gradient 경로 중 최적의 경로들을 찾아 강화(reinforcement)한다. Directed Diffusion의 단순화된 동작 구조는 <그림 2-1>과 같다.



(a) Interest propagation (b) Initial gradients set up (c) Data delivery along reinforced path

<그림 2-1> Directed Diffusion의 단순화된 구조
<Fig. 2-1> Simplified Schematic for Directed Diffusion

(1) 네이밍 규칙

Directed Diffusion은 interest와 data의 내용을 표현하기 위해 특정 속성을 명시하고 그 속성과 속성에 대응하는 값의 쌍으로 메시지를 구성하는 네이밍 규칙을 사용한다. <그림 2-2>는 사용자가 원

하는 내용을 네이밍 규칙에 맞춰 변환시킨 interest와 interest의 목표지역 내에 속한 노드에서 응답한 event 메시지의 한 예이다.

```
// interest
type=wheeled vehicle        // detect vehicle location
interval=10                 // send event every 10 ms
duration=10                 // for the next 10 s
rect=[-100,100,200,400]     // from sensors within rectangle

// event
type=wheeled vehicle        // type of vehicle seen
instance=truck              // instance of this type
location=[125,220]          // node location
intensity=0.6               // signal amplitude measure
confidence=0.85             // confidence in the match
```

<그림 2-2> interest와 event 메시지의 예

<Fig. 2-2> Example of Interest and Event Message

(2) interest와 gradient

싱크 노드는 위의 네이밍 규칙으로 interest 메시지를 생성하고 주기적으로 이웃 노드들에게 브로드캐스트한다. 초기에 확산되는 interest 메시지는 싱크 노드가 원하는 데이터의 존재를 탐색하는 것이 목적이다. 따라서 interval 속성의 값을 1초 정도로 설정하여 목표지역 내 센서 노드가 센서를 동작시키는 비율을 매우 낮게 한다. interest를 수신한 노드는 interest를 보낸 노드로 gradient를 계산하고 gradient 캐쉬에 이를 저장한다. 이와 동시에 interest 메시지를 분석하여 자신이 명시된 목표지역 내에 있는지를 검사한다. 검사결과 자신이 목표지역 내에 있다면 센서를 동작시켜 얻어진 데이터를 event 메시지로 변환하여 interest 메시지를 보낸 노드에게 gradient 경로를 따라 다시 전송한다. 검사결과 자신이 목표지역 내에 있지 않다면 interest를 다시 브로드캐스트하고 interest 캐쉬에 입력한다. 이때, 각 노드는 interest 메시지의 timestamp 속성을 이용해서 메시

지의 지연 시간도 함께 저장한다. 이 지연 시간은 뒤에 경로 강화의 척도로 사용된다.

(3) 데이터 전파와 경로 강화

interest를 수신한 센서 노드들은 센서를 동작시켜 데이터를 수집한 후 event 메시지를 생성하여 interest 메시지를 전달한 노드에게 즉 생성된 gradient 경로를 따라 유니캐스트한다. 그러나 interest 메시지가 브로드캐스트되어 여러 경로를 통해 전파되기 때문에 event 메시지 또한 여러 노드에게 유니캐스트되어 브로드캐스트와 같은 결과를 얻게 된다. 싱크 노드는 event 메시지를 수신하면 그 내용을 검사하고 자신이 요구하는 내용이면 동일한 event 메시지가 전달된 여러 gradient 경로들 중 가장 지연시간이 짧은 즉 높은 전송률을 보이는 경로를 선택해서 interval 속성의 값을 작게 하여 interest 메시지를 전송한다. 이 interest 메시지가 목표지역 내 소스 노드에 도착하면 센서의 동작 주기가 짧아지며 더욱 정확한 데이터를 측정하여 싱크 노드에게 전송하게 되고 이러한 과정을 경로 강화라고 한다.

2.1.3 평가

Directed Diffusion은 속성과 값의 쌍으로 이루어진 네이밍 모델을 도입하여 메시지를 형식화함으로써 각 소스 노드에서 질의메시지의 내용과 감지한 데이터를 비교하는데 일정한 규칙을 제공하여 매칭 룰(matching rule)을 간편화 시켰다. 뿐만 아니라 탐색용 interest와 event를 이용하여 정확한 데이터를 판별하고 품질이 좋은 전송 경로를 선택할 수 있는 방법을 제시하였다.

그러나 경로 강화를 위해서는 같은 목표지역으로 지속적인

interest와 event가 오가야하고 이 interest와 event가 브로드캐스트 되기 때문에 그만큼 많은 에너지를 낭비하게 된다. 또한 각 노드에서는 수신한 interest와 그에 대한 gradient를 보관해야 하며 이 양은 이웃 노드의 수에 비례하므로 노드의 밀집도가 높고 제한된 자원을 사용하는 WSN 환경에서는 많은 위험 요소를 안고 있다고 생각된다.

2.2 GEAR

2.2.1 GEAR의 개요

GEAR(Geographical Energe-Aware Routing)는 WSN에서 목표지역으로 질의메시지를 라우팅하기 위해 제안된 프로토콜이다. 이름에서 알 수 있듯이 지리적으로 이웃을 선택하는 학습 알고리즘과 에너지 인식 기법을 사용하며 다음의 네 가지 가정 사항을 가진다[10].

- 각 센서는 고정되어 있다.
- 각 패킷은 명시된 목표지역을 가진다.
- 각 노드는 자신의 위치와 잔류에너지 레벨을 알고 있으며 간단한 안부 패킷을 통해 이웃의 위치와 잔류에너지 레벨을 알 수 있다.
- 각 노드는 양방향 링크를 가진다.

GEAR는 '목표지역으로 패킷 전달'과 '목표지역 내에서 패킷 전달'의 두 단계로 동작한다. 목표지역으로 패킷을 전달할 때는 이웃 노

드의 지리정보와 잔류에너지를 이용하여 다음 홉을 선택하는 학습 기법을 사용한다. 이 때, 다음의 두 가지 경우를 고려한다.

- 자신보다 목적지에 더 가까운 이웃이 존재할 경우
  GEAR는 모든 이웃들 중에서 목적지에 더 가까이 있는 이웃을 다음 홉으로 선택한다.

- 모든 이웃이 자신보다 목적지에서 더 멀리 떨어져 있을 경우
  이럴 경우 망에 구멍(hole)이 생기며 GEAR는 이웃들 중 지리정보와 잔류에너지를 이용하여 계산한 특정비용이 가장 작은 노드를 다음 홉으로 선택한다.

목표지역 내에서 패킷을 전파할 때는 대부분의 경우 반복 지리적 전달(Recursive Geographic Forwarding) 알고리즘을 사용하고, 센서 노드의 밀도가 낮을 경우 제한적 플러딩(Restricted Flooding)을 사용한다.

2.2.2 GEAR의 동작

(1) 목표지역으로 패킷 전달

GEAR는 질의메시지를 목표지역으로 전달할 때 다음 홉을 선택하기 위해 지리정보와 잔류에너지를 이용하여 학습비용과 예측비용을 산출한다. 예를 들어, 노드 N이 중심이 D인 목표지역 R로 패킷 P를 전달한다고 가정해보자. 프로토콜의 목적이 라우팅에 소비되는 에너지를 네트워크 전체로 분산시켜 모든 노드에서 소비되는 에너지 분포를 고르게 하는 것이기 때문에, 노드 N은 패킷 P를 받자마자 목표지역 R을 향해 P의 경로를 정함과 동시에 자신의 모든 이

웃 노드에서 소비되는 에너지의 균형도 유지해야 한다. 이때 노드 N은 자신의 이웃들 중 학습비용 $h(N_i, R)$가 가장 작은 이웃 N_i를 다음 홉으로 정함으로써 이 목적을 달성한다.

네트워크를 구성하는 각 노드를 편의상 N이라고 했을 때, 각 노드 N은 목표지역 R에 대한 학습비용 $h(N, R)$를 유지한다. 그리고 모든 노드는 자신의 이웃 N_i의 학습비용 $h(N_i, R)$도 가진다. 만약 임의의 노드가 자신의 이웃 N_i의 학습비용 $h(N_i, R)$를 가지고 있지 않다면, 그에 대한 기본 값(default value)으로 예측비용 $c(N_i, R)$를 계산하여 유지한다. 예측비용 $c(N_i, R)$는 식 (2.1)과 같다.

$$c(N_i, R) = \alpha d(N_i, R) + (1 - \alpha) e(N_i) \tag{2.1}$$

식 (2.1)에서 α는 가중치이며 0과 1 사이의 값으로 조절할 수 있다. $d(N_i, R)$는 N의 모든 이웃 중, 지역 R의 중심 D에서 가장 멀리 떨어져 있는 노드 N_i와 D 사이의 거리이다. $e(N_i)$는 N의 모든 이웃 중, 에너지 소비가 가장 큰 노드 N_i에서 소비된 에너지이다.

노드 N은 자신의 이웃 중 학습비용 혹은 예측비용이 가장 작은 노드 $N_{\min}$을 다음 홉으로 뽑은 후, 자신의 학습비용 $h(N, R)$를 $h(N_{\min}, R) + c(N, N_{\min})$로 갱신한다. 여기서 $c(N, N_{\min})$는 N에서 $N_{\min}$으로 패킷을 전송하는데 드는 비용으로서, N에서 $N_{\min}$까지 거리와 N의 잔류 에너지 레벨을 결합한 조합 함수이다. 지금까지의 내용을 정리하면 다음과 같다.

- 각 노드는 패킷을 전송할 때 모든 이웃들의 학습비용을 비교한다.
- 만약 학습비용을 알 수 없는 이웃이 있다면 예측비용을 계산하

여 대체한다.

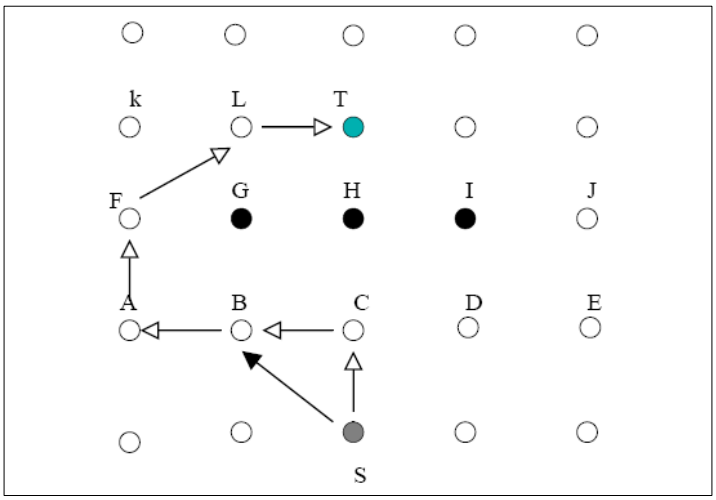
- 학습비용이 가장 작은 이웃을 다음 홉으로 선택하고 자신의 학습비용을 갱신한다.

GEAR는 학습비용과 예측비용을 이용하여 다음 홉을 정하는 기준으로 삼고, 다음 홉이 결정될 때마다 각 노드의 학습비용을 갱신하도록 함으로써 각 노드의 지리정보와 잔류에너지 레벨을 경로 선택에 적극적으로 반영할 수 있도록 하였다. 위의 과정을 거쳐 노드 N은 자신의 각 이웃에 대한 학습비용 $h(N,R)$ 혹은 이에 대한 기본값인 예측비용 $c(N,R)$를 모두 가지므로 이 비용들을 기준으로 라우팅을 시작한다.

앞서 기술했듯이, GEAR는 목표지역으로 패킷을 전달할 때 자신보다 목적지에 더 가까운 이웃이 존재할 경우와 그렇지 않은 경우로 나누어 라우팅을 수행한다. 더 가까운 이웃이 있을 때는 그리디(greedy) 알고리즘을 적용하여 이웃들 중 목표지역에 더 가까운 노드를 고른 후, 그들의 학습비용을 비교하여 가장 작은 노드를 다음 홉으로 결정한다. 이와 반대로, 더 가까운 이웃이 없을 때는 조금 더 복잡한 처리과정을 가진다. 더 가까운 이웃이 없을 때는 송신 노드와 목적 노드 사이에 구멍(hole)이 생기기 때문이다. 이 구멍은 송신 노드와 목적 노드 사이에 데이터를 전달하는 다리역할을 해주는 노드가 없다는 것을 뜻한다. 따라서 송신 노드가 목적 노드에 패킷을 전송하기 위해서는 구멍을 우회해야만 한다.

GEAR는 학습비용의 갱신 규칙을 통해 구멍을 우회하는 방법을 제안한다. 예를 들어, <그림 2-3>과 같은 격자형 토폴로지를 가정해 보자. 가장 가까운 두 이웃 사이의 거리는 1이고 각 노드는 자신을 둘러싼 여덟 이웃과 통신할 수 있다. 검은색으로 표시된 G·H·I는 에너지를 모두 소비한 노드이므로 패킷을 전달할 수 없다. 이때, 노

드 S가 노드 T로 패킷을 보내기 위해 경로를 탐색한다면 다음과 같은 시나리오가 수행된다.



<그림 2-3> 구멍을 우회하기 위한 경로 학습
<Fig. 2-3> Learning Routes around Holes

최초 시간 0일 때, 노드 S는 자신과 통신할 수 있는 모든 이웃 중 자신보다 더 가까운 이웃인 B·C·D를 발견한다. B·C·D의 학습비용은 각각 $h(B,T)=c(B,T)=\sqrt{5}$, $h(C,T)=c(C,T)=2$, $h(D,T)=c(D,T)=\sqrt{5}$이다. 노드 S는 학습비용을 비교하여 노드 C를 다음 홉으로 정하고 패킷을 전송한다.

시간 1일 때, 노드 C는 자신보다 목적지에 더 가까운 이웃 G·H·I가 모두 통신 불능이므로 자신이 구멍에 봉착했다는 것을 알게 된다. G·H·I 외의 모든 노드가 자신보다 T에서 멀리 있으므로, 학습비용이 최소인 B·D 중 다음 홉을 선택한다. 이때, 노드 B·D의 학습비용이 같으므로 노드 식별자의 순서 등을 이용한 임의의 방법으로 다음 홉을 정하고 패킷을 전송한다. 그런 다음 노드 C는 자신의 학습비용을 $h(B,T)+c(C,B)=\sqrt{5}+1$로 수정한다.

시간 2일 때, 노드 S는 다음 패킷을 전송한다. 이때, B·C·D의 학습비용은 각각 $h(B,T)=\sqrt{5}$, $h(C,T)=\sqrt{5}+1$, $h(D,T)=\sqrt{5}$ 이므로 노드 S는 학습비용이 작은 B·D 중 임의의 방법을 통해 다음 홉을 선택하여 구멍을 우회하게 된다.

(2) 목표지역 내에서 패킷 전달

패킷은 목표지역에 도착하기 전까지 앞서 기술한 방법을 통해 전달되지만 목표지역 내에서는 반복 지리적 전달과 제한적 플러딩에 의해 전달된다. GEAR는 대부분의 경우 반복 지리적 전달을 통해 패킷을 전달한다.

반복 지리적 전달은 목표지역을 다수의 하위지역(sub-region)으로 나누고 각 지역으로 패킷의 사본을 유니캐스트(unicast)하며 각 하위지역에서 제일 처음 패킷을 수신한 노드는 자신이 속한 하위지역을 또 다시 다수의 하위지역으로 나누어 패킷의 사본을 유니캐스트하는 방식으로써 이러한 반복적인 전달은 나누어진 하위지역이 완전히 비워질 때까지 즉, 하위지역 내에 단 하나의 노드도 없을 때까지 계속된다. 따라서 네트워크를 구성하는 노드의 밀도가 낮을 경우, 노드의 통신 범위가 하위지역 전체에 이르지 못하여 하위지역내 노드의 유무를 파악하지 못하는 경우가 발생하며 패킷의 전달 동작은 종료되지 못하고 TTL(Time To Live)이 다할 때까지 하위지역 주변을 떠돌게 된다. 패킷의 전달이 종료되지 못하고 TTL이 다할 때가지 떠돌게 되면 그만큼 각 노드에서 소모되는 에너지가 늘어나게 되므로 GEAR는 네트워크의 밀도가 낮을 경우, 반복 지리적 전달 대신 제한적 플러딩의 사용을 제안한다.

제한적 플러딩은 반복 지리적 전달과 달리 목표지역내에 패킷을 전달할 때 플러딩을 사용하지만 과도한 에너지의 소모를 막기 위해 패킷의 홉 카운트를 한 홉이나 두 홉 정도로 제한하는 방식이다. GEAR는

패킷을 전달하는 노드의 이웃의 개수를 통해 네트워크의 밀도를 예측한다. 그리고 특정 값을 임계치로 정해 놓고 노드의 이웃이 이 값보다 많으면 반복 지리적 전달을 사용하고 그렇지 않으면 제한적 플러딩을 사용한다.

2.2.3 평가

GEAR는 노드의 위치 정보와 잔류에너지 레벨을 활용하여 네트워크의 에너지 소모를 균형 있게 하여 네트워크의 생존시간을 증가시킬 수 있었다. 지역적으로 산출한 특정비용을 고려하여 목적지로 메시지를 전송할 수 있는 최적의 결정을 내린다. 그리고 효율적인 경로를 학습하고 망의 구멍을 우회하는 능력도 가진다. 노드의 위치와 잔류에너지, 이웃 노드의 위치와 잔류에너지를 계속 업데이트해야 하는 추가 작업을 필요로 하지만 WSN에서 노드의 위치 정보는 응용에 필요한 정보이기 때문에 이를 경로 검색에 이용하여 데이터 전송량을 효과적으로 줄이는 결과를 보였다.

그러나 GEAR는 이동성을 고려하지 않았다. 이것은 망의 구멍을 우회하는 규칙에서 극명하게 드러난다. 본 논문에서 직접적으로 기술하진 않았지만, GEAR를 제안한 논문에서는 <그림 2-3>에서 구멍을 감지한 노드 C의 학습비용이 노드 B·D의 학습비용보다 큰 값을 가지기 위해서는 상당한 양의 패킷이 전송되어야 하고 그만큼의 시간이 요구된다는 실험 결과를 명시하였다. 따라서 동적인 대규모의 네트워크에서 GEAR는 결코 좋은 성능을 낼 수 없을 것이다.

제 3 장 제안된 라우팅 프로토콜

본 장에서는 제안된 라우팅 프로토콜의 개요와 동작에 대해 기술한다. 본 논문은 시뮬레이션 수준에서 프로토콜의 성능을 평가하므로 프로토콜을 위한 알고리즘에 사용한 위치좌표는 실제 경·위도가 아닌 단순한 x·y 좌표로 추상화 시켰다. 또한 프로토콜의 설계/구현에 사용되는 함수들은 NS-2 시뮬레이터에서 제공하는 NR (Network Routing) API 함수에 의존적임을 밝힌다. 프로토콜은 Redhat LINUX 9.0(kernel ver. 2.4.20-8)에서 C++ 프로그래밍 언어를 사용하여 설계/구현하였다[11].

3.1 제안된 프로토콜의 개요

본 논문에서 제안된 프로토콜은 다음의 네 가지 가정을 전제로 한다.

- 모든 센서 노드는 이동성을 가지지만 그 이동은 빈번하지 않다.
- 센서 필드 내의 모든 센서 노드에서 전송하는 데이터의 목적지는 싱크 노드이다.
- 싱크 노드는 고정되어 있다.
- 센서 필드 내의 모든 센서 노드는 자신과 싱크 노드의 위치좌표를 알고 있다.

위의 가정 사항들은 WSN의 요구기반 모델과 일반적인 응용 모델을 기반으로 한다. WSN의 일반적인 응용 모델은 특정 지역을 감시하기 위해 운용되므로 센서 노드의 이동은 그리 빈번하지 않다.

싱크 노드는 각 센서 노드에서 감지한 환경데이터를 통합해서 관리자에게 넘겨주는 역할을 하므로 센서 노드에서 획득한 데이터의 목적지는 싱크 노드라고 가정한다. 또한 싱크 노드는 호스트 PC의 시리얼이나 USB 단자에 접속되어 있는 경우가 대부분이므로, 싱크 노드는 고정되어 있으며 사용되는 에너지는 제한적이지 않다.

마지막으로, 서론에서 언급했듯이 GPS 모듈을 내장한 센서 노드가 이미 개발된 상태이므로 모든 센서 노드가 자신과 싱크 노드의 위치 정보를 알 수 있다는 가정 또한 타당하다.

제안된 라우팅 프로토콜의 동작을 기술하기 전에 노드의 이동성에 대해 논의할 사항이 있다. 이는 노드의 이동성이 WSN의 주요 특징 중의 하나임에도 불구하고 WSN을 위해 제안된 대부분의 라우팅 프로토콜이 노드의 이동성을 고려하지 않는 이유에 대한 설명이기도 하다.

WSN은 1996년 미국의 UC Berkeley에서 스마트 더스트(Smart Dust) 프로젝트를 추진하면서 본격적으로 연구되기 시작한 분야이다. 스마트 더스트는 그 이름이 말하듯 똑똑한 먼지를 만드는 것을 목적으로 하는 프로젝트이다. 먼지만큼 작고 가벼운 센서 노드를 만들어 주변 환경에 배포하고 각 센서 노드가 감지한 환경데이터를 활용하여 인간 생활의 편의를 도모할 뿐만 아니라 위급·응급 상황이 발생했을 때, 신속히 대처하는 것을 최종 목표로 한다[3].

스마트 더스트를 실현하기 위해 해결되어야하는 많은 사항들 중 센서 노드를 관리하기 위한 운영체제에 대한 연구가 2000년에 UC Berkeley에서 TinyOS 프로젝트라는 이름으로 시작되었다. 이 운영체제의 이름은 TinyOS라고 하며 TinyOS의 실험을 위해 mote라고 부르는 임베디드 보드를 제작하고 이를 센서 노드로 사용하여 무선 네트워크를 구축하였다. 이때의 데모가 현재 WSN 응용의 시초라고 할 수 있다[4][12][13].

WSN은 스마트 더스트를 최종 목표로 하여 연구가 시작되었고 지금도 활발히 연구가 진행되고 있지만, 현재 응용되고 있는 WSN은 스마트 더스트를 위한 연구의 성과가 얼마나 실제상황에 적용가능한 지를 실험해보기 위한 테스트베드로써 탄생한 것이다. 이동성의 관점에서 보면, 스마트 더스트에서 정의한 센서 노드의 이동은 굉장히 활발하다. 이는 스마트 더스트가 지향하는 센서 노드가 먼지처럼 작고 가볍기 때문에 당연한 것이다. 하지만 현재 응용되고 있는 WSN의 노드는 <표 3-1>과 같은 스펙을 가지고 있으며 응용분야가 주로 감시·관찰 임무에 집중되어 있으므로 노드의 이동이 활발하지 않다. <표 3-1>은 UC Berkeley에서 개발한 mote들 중 MICA 시리즈를 개량하여 국내 전자부품연구원(KETI; Korean Electronics Technology Institute)에서 개발하고 (주)맥스포에서 제품화한 TIP 시리즈 중 50CM 보드의 스펙이다[14][15].

<표 3-1> 맥스포(주) TIP 50CM의 스펙
<Table 3-1> Specification of Maxfor Inc. TIP 50CM

Processor	16bit RISC, 8MHz(MSP430)
Memory	256KB Program Flash
Operating System	TinyOS
Multi-Channel Radio	2.4GHz
Data Rate	250Kbps
Sensor	Temperature, Humidity and Light
Network	Multi Hop and Ad-Hoc
Interface	USB(UART)
Size	68×29mm
Power	3.0-3.3V
Range	70m in Lab.

현재 WSN에서 야생동물의 이동을 추적하는 등의 특별한 응용을 제외하고 센서 노드가 이동하는 경우는 서론에서 언급하였듯 자연 재해에 의한 예기치 못한 경우밖에 없다. 이런 이유로 현재 WSN의 응용을 위한 라우팅 프로토콜은 대부분 이동성을 고려하지 않는다. 따라서 기존의 프로토콜은 동적인 환경이라면 노드가 빈번히 이동하지 않는다 하더라도 정적인 환경에 비해 그 성능이 저조해진다. 그러나 본 논문에서는 노드의 이동성을 WSN의 특징 중 자가-구성, 고장 감내와 함께 라우팅 프로토콜에서 반드시 고려되어야 하는 사항으로 간주하며 현재 WSN의 응용에서 발생하는 예기치 못한 노드의 이동, 즉 노드가 빈번히 이동하지 않는 환경에서라면 노드가 이동하더라도 프로토콜의 성능에 큰 영향이 없도록 설계하였다 [16][17].

본 논문에서 제안된 프로토콜은 방향성 설정, 다음 홉 선정, 메시지 전송의 세 가지 동작 단계를 가진다. 방향성 설정과 다음 홉 선정 단계는 각 노드가 메시지 전송 단계에서 질의메시지나 데이터 메시지를 발생하기 전에 수행하는 일종의 준비동작의 역할을 하며 이 두 동작 단계를 통해 노드의 이동성, 자가-구성, 고장 감내의 특성을 지원하도록 하였다.

3.2 제안된 프로토콜의 동작

3.2.1 방향성 설정 단계

방향성 설정 단계는 센서 노드들이 배포되고 난 후, 제일 먼저 수행하는 동작 단계이다. 각 센서 노드들은 자신과 싱크 노드의 위치 좌표를 이용해 싱크 노드로의 방향성을 설정한다. 이때 사용되는 방

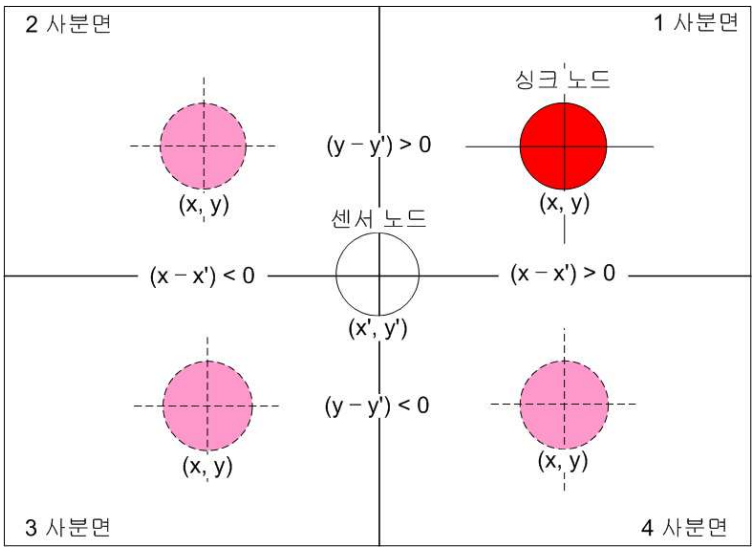
향성 정의 알고리즘은 이차원 좌표 평면에서 한 점을 기준으로 다른 한 점의 상대적인 위치를 1, 2, 3, 4 사분면 중의 하나로 추상화시키는 간단한 수식을 이용하여 각 노드의 방향성을 정의한다.

(1) 방향성 정의 알고리즘

본 논문에서 제안하는 프로토콜의 가장 큰 특징은 각 센서 노드의 위치좌표를 이용해서 패킷의 목적지(싱크 노드에서는 목표 지역으로, 각 센서 노드에서는 싱크 노드로)로의 방향성을 정의하여 방향성에서 벗어난 노드의 신호발생을 억제함으로써 네트워크에서 소모되는 에너지를 감소시키는 것이다.

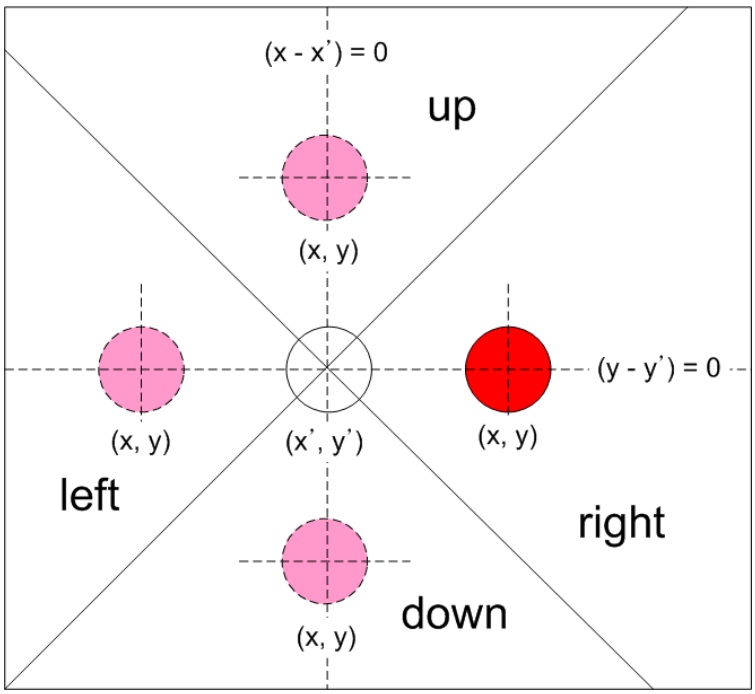
노드에서 사용되는 에너지는 신호를 송·수신할 때 가장 많이 소모된다. 센서 노드에서 사용하는 안테나는 전 방향 안테나(omni-directional antenna)이므로 일단 신호가 발생하면 통신 반경 내에 있는 다른 센서 노드들은 그 신호를 수신할 수밖에 없다. 따라서 신호를 수신한 노드에게 수신한 신호를 다른 노드에게 전달할 것인지, 않을 것인지에 대해 결정하게 함으로써 통신에 사용되는 에너지 소모를 줄일 수 있다. Directed Diffusion과 GEAR에서는 단순한 플러딩을 사용하거나, 노드간 거리와 잔류 에너지 레벨을 기반으로 계산한 비용 값을 이용했지만 본 프로토콜에서는 목적지까지의 방향성을 정의하고 이것을 의사 결정의 근거로 이용한다.

방향성 정의는 각 센서 노드를 원점으로 하는 사분면을 형성함으로써 이뤄진다. 예를 들어 싱크 노드의 위치좌표를 (x, y), 각 센서 노드의 위치좌표를 (x′, y′)이라 했을 때, 방향성 정의 알고리즘은 <그림 3-1>과 같다. 그리고 싱크 노드의 방향은 (x-x′)과 (y-y′)이 양수인지 음수인지에 따라 각 센서 노드를 원점으로 하는 평면의 1, 2, 3, 4 사분면으로 추상화된다.



<그림 3-1> 방향성 정의 알고리즘의 개념

<Fig. 3-1> Concept of Direction Definition Algorithm



<그림 3-2> 예외상황의 처리

<Fig. 3-2> Processing of Exception

<그림 3-1>에서 (x-x′) = 0 이거나 (y-y′) = 0 인 경우가 발생할 수 있다. 이것은 두 노드가 같은 X 좌표 혹은 Y 좌표 값을 가진다는 것을 뜻한다. 실세계로 따지자면 같은 경·위도 좌표를 가진다는 것을 뜻한다. 이럴 경우는 목적지의 방향성을 사분면으로 추상화시킬 수 없다. 따라서 본 논문에서는 <그림 3-2>와 같이 각 센서 노드의 상·하·좌·우 90° 의 각도로 싱크 노드의 방향을 추상화시키고자 한다. <그림 3-1>과 <그림 3-2>의 알고리즘을 C++ 언어로 표현한 것은 <그림 3-3>과 같다.

```
// 방향성 정의 알고리즘
void TestSenderApp::DirectionDef() {
  bool up = down = left = right = false;
  float subtraction_x = x - xp;
  float subtraction_y = y - yp;

  if(subtraction_x) {
    right = true;
    if(subtraction_y) up = true;
    else if(subtraction_y < 0) down = true;
  }
  else if(subtraction_x == 0) {
    if(subtraction_y) up = true;
    else if(subtracion_y < 0) down = true;
  }
  else {
    left = true;
    if(subtraction_y) up = true;
    else if(subtraction_y < 0) down = true;
  }
}
```

<그림 3-3> C++로 나타낸 방향성 정의 알고리즘
<Fig. 3-3> Direction Definition Algorithm in C++

각 센서 노드와 싱크 노드는 위의 방식으로 true 값을 할당받은 bool 타입 변수가 어떤 것이냐에 따라 <표 3-2>와 같이 목적지의 방향을 설정할 수 있다.

<표 3-2> 변수에 따른 방향성 설정
<Table 3-2> Setting Direction by Variables

변수	사분면				상	하	좌	우
	1	*2*	*3*	*4*	90°	90°	90°	90°
up	O	O			O			
down			O	O		O		
left		O	O				O	
right	O			O				O

방향성 정의 알고리즘은 싱크 노드에서 목표지역으로 방향을 설정하는 경우와 각 센서 노드에서 싱크 노드로 방향을 설정하는 경우로 나누어 적용된다.

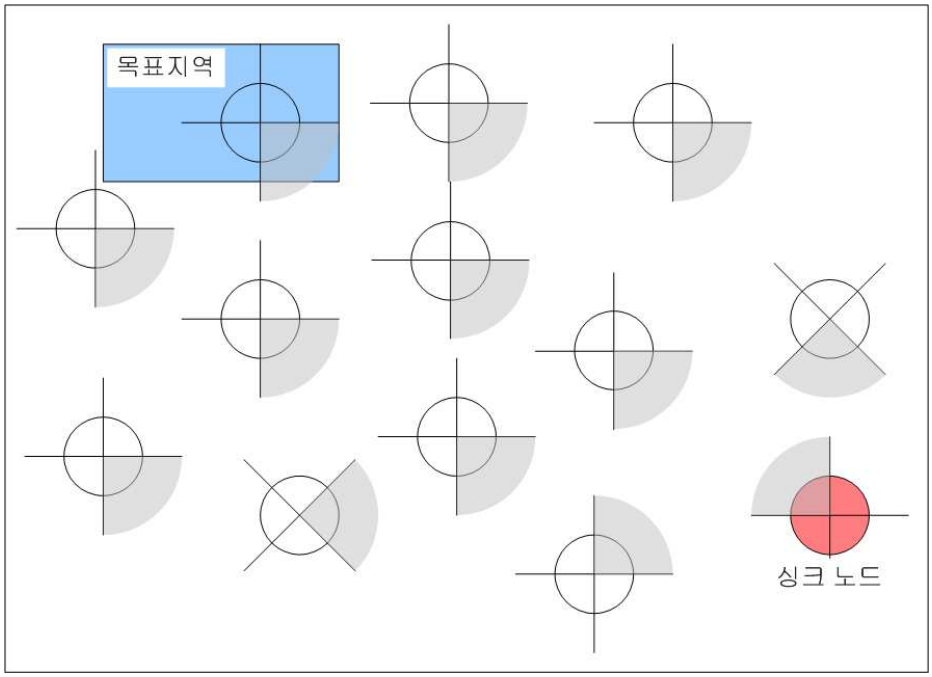
- 싱크 노드에서 목표지역으로 방향을 설정하는 경우

  싱크 노드에서 목표지역으로 방향을 설정하는 경우, 방향성 정의 알고리즘은 질의메시지에 명시된 목표지역의 값을 이용한다. 본 프로토콜에 적용한 목표지역은 2 차원 좌표 체계를 나타내며, Directed Diffusion에서 사용된 목표지역의 형식과 달리 일반적인 프로그래밍 언어의 문법에서 사용하는 X 좌표, Y 좌표, 너비, 높이의 네 가지 인자를 이용해서 사각형으로 이뤄지는 특정지역의 경계면을 표시하도록 하였다. 따라서 목표지역의 처음 두 인자(X 좌표, Y 좌표)와 싱크 노드의 위치 좌표를 방향성 정의 알고리즘에 대입하면 특정 지역으로의 방향을 설정할 수 있다.

• 각 센서 노드에서 싱크 노드로 방향을 설정하는 경우

각 센서 노드에서 싱크 노드로 방향을 설정하는 경우, 본 알고리즘은 <그림 3-1>과 완전히 동일하게 동작한다. 각 센서 노드는 입력된 싱크 노드의 위치좌표와 자신의 위치좌표를 이용해서 방향을 설정한다.

센서 노드가 배포되고 방향성 설정 단계를 수행한 후 각 센서 노드에서 방향성이 설정된 모습을 <그림 3-4>에 나타내었다.



<그림 3-4> 각 노드에서 방향성이 설정된 모습

<Fig. 3-4> Status after Setting Direction at Each Node

각 센서 노드는 센서 필드에 배포되고 난후 스스로 네트워크를 구성하여 <그림 3-4>와 같이 자신들의 목적지로 방향성을 설정하고 다음 홉 선정 단계를 시작한다.

3.2.2 다음 홉 선정 단계

방향성 설정이 완료되고 나면, 각 노드에서는 자신의 방향성에 입각한 노드들 중 통신 범위 내에서 가장 멀리 떨어져 있는 노드를 식별한다. 실제 센서 노드의 통신 반경이 최대 70m 정도이고 각 노드가 배포될 때 무작위로 뿌려진다는 점을 고려했을 때, 통신 반경 내에서 가장 가까운 노드를 다음 홉으로 선택한다면 각 노드에서 싱크 노드까지의 경로 상에 있는 거의 모든 중간 노드들이 신호를 수없이 중복 수신하게 된다는 결론이 나온다. 이는 불필요한 에너지의 낭비를 초래하고 결과적으로 전체 네트워크의 수명을 단축시킬 것이다. 따라서 비록 통신에 소모되는 에너지가 두 노드간 거리의 제곱에 비례한다 하더라도 통신 반경 내에서 가장 멀리 떨어져 있는 노드를 다음 홉으로 선택하여 신호원이 되는 노드를 목적지로 최대한 빠르게 이동시킴으로써 중간 노드가 신호를 중복 수신하는 것을 막고 소모되는 에너지를 줄인다.

통신 반경 내 가장 멀리 떨어져 있는 노드를 식별하기 위해 각 노드는 간단한 포맷의 헬로 패킷을 이용한다. <표 3-3>에 헬로 패킷의 포맷을 나타내었고 이를 C++ 언어로 작성한 것은 <그림 3-5>와 같다.

<표 3-3> 헬로 패킷의 포맷
<Table 3-3> Format for Hello Packet

PID (Packet ID)	*HL (Hello Location)*	*RL (Response Location)*	*Direction*
H (Hello)	(x, y)	empty	up, down, left, right
R (Response)		(x′, y′)	

```
// Hello Packet's Format
class HelloPacket {
public:
  HelloPacket();
  void setPID(char pid);
  void setHL(float x, float y);
  void setRL(float x, float y);
  void setDir(bool up_dn, bool rgt_lft);
private:
  char pid;
  float hlx, hly;
  float rlx, rly;
  bool up, down, right, left;
};
```

<그림 3-5> C++로 작성한 헬로 패킷

<Fig. 3-5> Format for Hello Packet in C++

헬로 패킷을 이용하여 다음 홉을 선정하는 알고리즘은 H(hello) 패킷 전송 노드와 R(response) 패킷 전송 노드로 나누어 적용된다.

• H 패킷 전송 노드

1. H 패킷을 전송하고 타이머를 동작시킨다.
2. 일정시간 대기 후 응답이 없으면 H 패킷을 다시 전송한다.
3. 응답이 있으면 타이머를 리셋하고 수신한 패킷의 PID 필드를 확인한다.
4. PID 필드의 값이 H인 경우, R 패킷 전송 노드 알고리즘을 3부터 수행한다.
5. PID 필드의 값이 R인 경우, HL과 Direction 필드의 값을 확인한다.
6. HL과 Direction 필드의 값이 자신이 전송한 H 패킷의 값과 동

일하다면, RL 필드의 값을 이용하여 노드 간 거리를 계산한다.

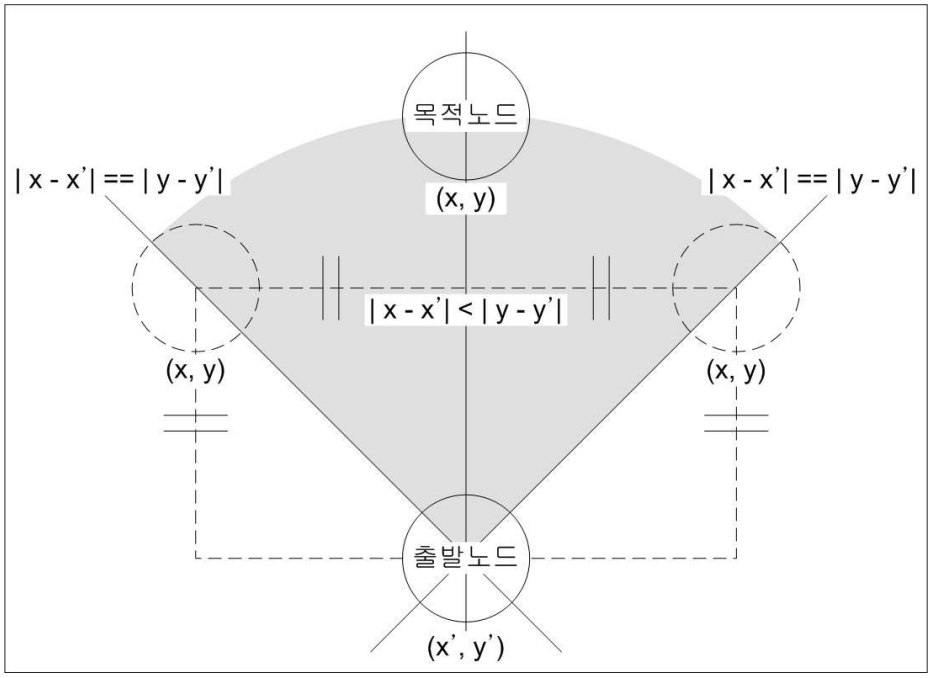
7. 계산한 거리 값을 캐쉬에 있는 값과 비교하여 클 경우, 캐쉬에 있는 값들을 새로운 RL과 노드 간 거리로 갱신한다.
8. 계산한 거리 값이 캐쉬에 있는 값보다 작을 경우 패킷을 폐기한다.

• R 패킷 전송 노드

1. 수신한 패킷의 PID 필드를 확인한다.
2. PID 필드의 값이 R인 경우 H 패킷 전송 노드 알고리즘을 5부터 수행한다.
3. PID 필드의 값이 H인 경우 HL 필드의 값을 확인한다.
4. HL 필드의 값과 자신의 위치 좌표를 이용해 H 패킷을 전송한 노드에 대한 자신의 상대적인 방향성을 계산하고 Direction 필드의 값과 비교한다.
5. Direction 필드의 값이 일치할 경우 패킷의 PID를 R로 바꾸고 RL 필드에 자신의 위치 좌표를 입력하여 패킷을 전송한다.
6. Direction 필드의 값이 일치하지 않을 경우 패킷을 폐긴한다.

위의 알고리즘에서 방향성을 확인하는 과정은 <표 3-2>에서 보인 바와 같이 방향성이 사분면으로 정의되어있는 경우와 그렇지 않은 경우로 나뉘어 수행된다. 방향성이 사분면으로 정의되어있는 경우는 두개의 bool 변수를 사용해서 사분면이 표시되므로 수신한 R패킷의 RL 필드에 저장된 위치좌표와 자신의 위치좌표를 방향성 정의 알고리즘에 대입하여 그 결과 값과 목적지의 방향성이 일치하는지를 비교하면 된다. 방향성이 사분면으로 정의되어있지 않은 경우는 하나의 bool 변수만 사용되므로 활성화되는 변수에 따라 상·하·좌·우 90°로 목적지의 방향을 추상화시킨다. <그림 3-6>은 변수 up이 활

성화되었을 때 적용되는 알고리즘을 나타낸 것이다.



<그림 3-6> 변수 up이 활성화되었을 때 방향성 설정
<Fig. 3-6> Setting Direction when Variable up is True

변수 up이 활성화되었다는 것은 출발 노드와 목적 노드의 X좌표가 동일하고 목적 노드의 Y좌표가 출발 노드의 Y좌표보다 크다는 것을 의미한다. 이때, 상 90°로 방향을 설정하려면 (x-x′)의 절대값이 (y-y′)의 절대 값보다 작아야 된다. 따라서 목적지의 방향성이 up일 때 다음 홉을 선정하기 위해서는 (y-y′) > 0 와 |x-x′| < |y-y′| 를 동시에 만족하는 노드 중 출발 노드에서 가장 멀리 떨어져 있는 노드를 다음 홉으로 선정한다. 나머지 다른 방향을 설정하는 것도 <그림 3-6>의 알고리즘과 동일하게 처리하며 상·하·좌·우로 방향성을 설정하기 위해 만족해야 하는 조건을 <표 3-4>에 나타내었다.

<표 3-4> 상·하·좌·우로 방향성 설정 조건
<Table 3-4> Setting Direction toward up·down·left·right

	$\|x-x'\| < \|y-y'\|$	$\|x-x'\| > \|y-y'\|$
$(y-y') > 0$	up 90°	
$(y-y') < 0$	down 90°	
$(x-x') > 0$		right 90°
$(x-x') < 0$		left 90°

각 센서 노드와 싱크 노드에서는 위에서 기술한 방식을 통해 목적지로의 방향성에 따라 다음 홉을 결정할 수 있다. 방향성 설정 단계와 다음 홉 선정 단계는 본 프로토콜이 WSN의 자가-구성, 고장 감내, 이동성을 지원하기 위한 중요한 수단이기도 하다. 이동성 지원에 대해서는 메시지 전송 단계에서 설명한다.

3.2.3 메시지 전송 단계

메시지 전송 단계는 다음 홉으로 질의메시지나 데이터 메시지를 전송하는 단계이다. 앞 절에서 기술한 방향성 정의와 다음 홉 선정의 두 단계를 통해 각 노드에서 정해진 다음 홉으로 메시지를 전송하며 각 메시지의 표현은 Directed Diffusion과 동일한 네이밍 방식을 그대로 사용한다. 이러한 내용은 이미 앞에서 기술되어 있는 내용이므로 본 절에서는 본 프로토콜이 어떻게 노드의 이동성을 지원하는지에 대해 기술한다[18][19].

MANET을 위한 프로토콜은 크게 테이블 기반(Table-driven) 방식과 요구 기반(On-demand) 방식으로 나뉜다. 테이블 기반 방식은 주기적으로 탐색 메시지를 발생시켜 이웃의 변화를 감지하는 방식

이고, 요구 기반 방식은 요청 메시지가 있을 때 다음 홉을 정하기 위해 이웃의 존재를 탐색하는 방식이다. 따라서 네트워크 환경이 동적이라면 테이블 기반 방식을 사용하고 그 반대의 경우라면 요구 기반 방식을 사용한다. 이러한 방법론은 WSN에도 그대로 적용될 수 있다. 본 논문에서 제안하는 프로토콜은 노드의 이동이 그리 빈번하지 않은 환경을 가정하므로 MANET의 요구 기반 방식을 이용하여 노드의 이동성을 지원하고자 한다[20].

예를 들어, 위치좌표가 (x, y)인 노드 A가 (x′, y′)의 위치로 이동하였을 때, 노드 A는 <표 3-5>와 같은 포맷의 헬로 패킷을 발생한다. PID의 M은 패킷을 발생한 노드의 위치가 수정되었음을 뜻한다. HL 필드에는 현재의 위치좌표를 입력하고 Direction 필드에는 이전 위치로의 방향성을 입력하여 패킷을 전송한다. M 패킷을 수신한 각 노드는 다음 홉 선정 단계를 다시 수행하고, 위치가 변경된 노드 A는 방향성 설정 단계와 다음 홉 선정 단계를 다시 수행한다.

<표 3-5> 노드가 이동하였을 때 발생하는 헬로 패킷의 포맷
<Table 3-5> Format for Mobility of Nodes

PID (Packet ID)	*HL (Hello Location)*	*RL (Response Location)*	*Direction*
M (Modify)	(x′, y′)	empty	up, down, left, right

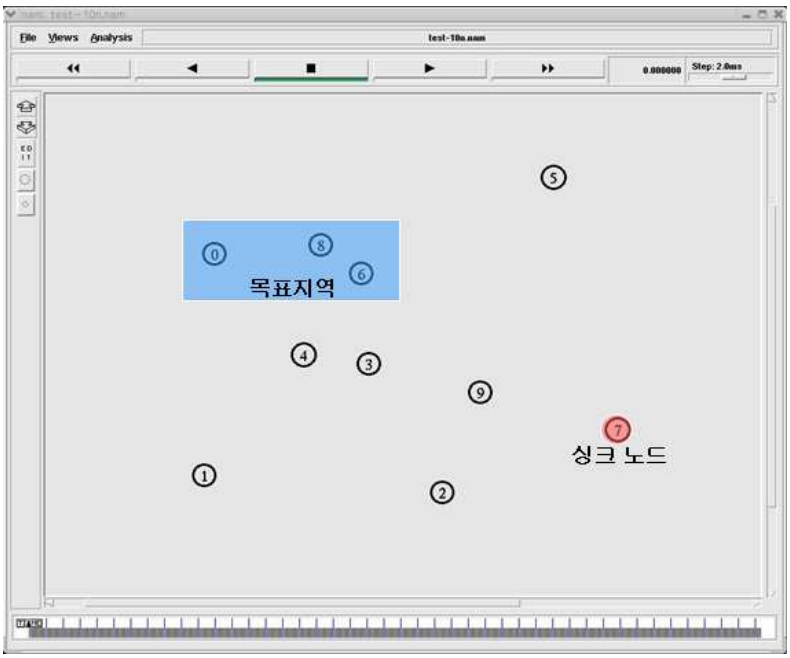
M 패킷은 이전의 위치로 방향이 설정되어있으므로 노드 A의 이전 위치에서 노드 A의 이웃이었던 노드들은 모두 M 패킷을 수신할 수 있다. 또한 M 패킷을 수신한 노드들은 목적지의 방향으로만 패킷을 전송하므로 다른 노드들이 불필요한 메시지를 발생시키는 것을 억제할 수 있다.

제 4 장 시뮬레이션 및 결과 분석

본 장에서는 방향성 정의 알고리즘의 동작을 검증하고 본 논문에서 제안한 프로토콜과 GEAR의 성능을 비교해본다. 시뮬레이션은 Redhat LINUX 9.0(kernel ver. 2.4.20-8)에서 NS-2(ver. 2.30) 시뮬레이터를 이용하였고, AWK 프로그래밍 언어를 이용하여 시뮬레이션 결과 데이터를 처리하고 Microsoft Windows XP Professional에서 Microsoft Office Excel 2003을 이용해 시각화 하였다[21][22].

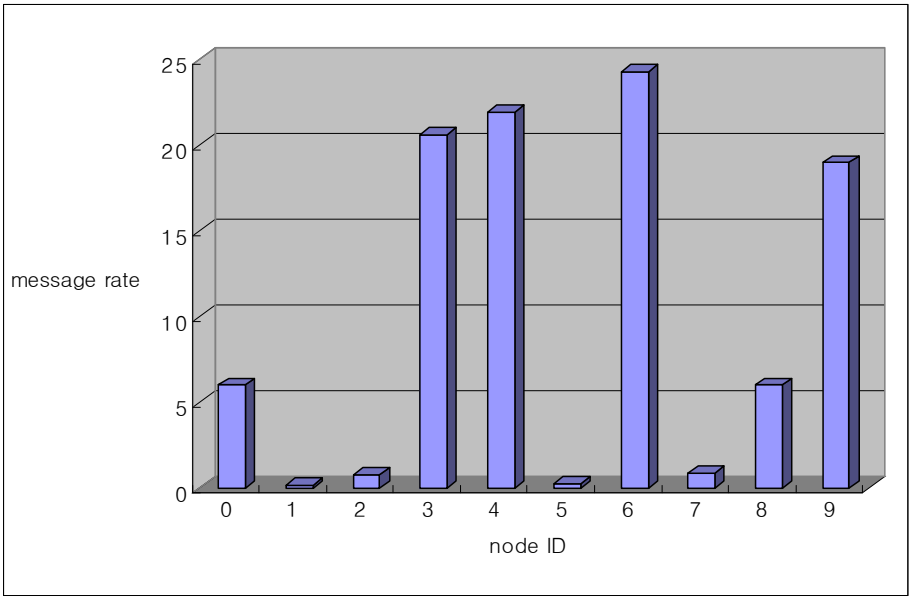
4.1 방향성 정의 알고리즘의 동작 검증

<그림 4-1>과 같이 10개의 노드로 네트워크를 구성하였다.



<그림 4-1> 방향성 정의 알고리즘의 동작 검증
<Fig. 4-1> Evaluation of Direction Definition Algorithm

각 노드는 직접 x·y 좌표를 입력하여 위치시켰다. 7번 노드를 싱크 노드로, 0·6·8번 노드가 속한 직사각형을 목표지역으로 정했을 때 알고리즘이 설계했던 데로 정확히 동작한다면 싱크 노드와 목표지역 사이에 위치한 3·4·9번 노드에서 송·수신되는 메시지의 양이 다른 노드에 비해 많아야 한다. 시뮬레이션 시간은 500초로 하고 총 10번을 반복하여 송·수신되는 메시지의 양에 변화가 있는지를 관찰하였다. 10번의 시뮬레이션을 거치는 동안 송·수신 메시지의 양에는 전혀 변화가 없었으며 각 노드를 거치는 송·수신 메시지 양의 비율은 <그림 4-2>와 같다.



<그림 4-2> 각 노드별 송·수신 메시지 양의 비율
<Fig. 4-2> Message Rate at Each Nodes

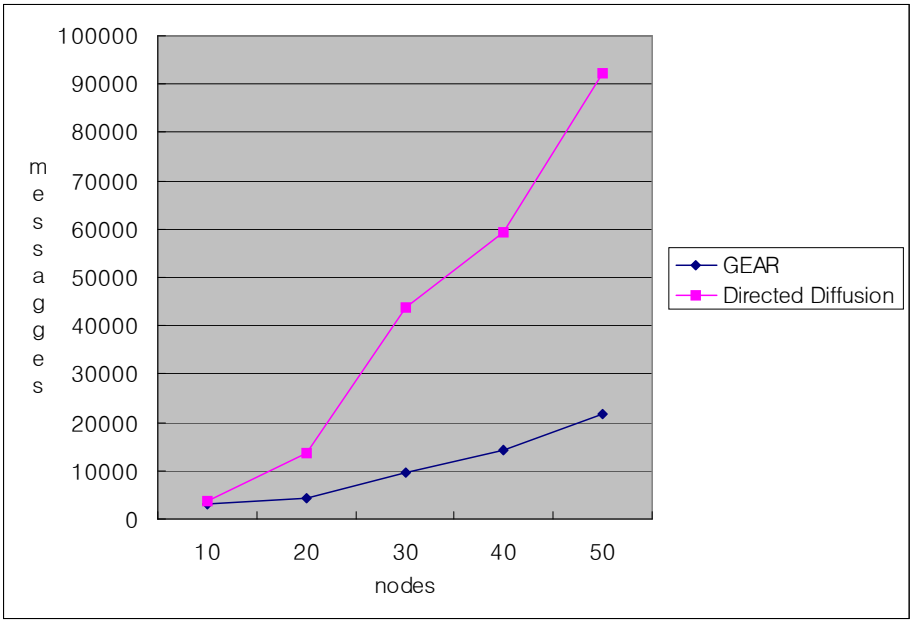
<그림 4-2>를 보면 전체적으로 예상했던 내용과 비슷한 결과를 보임으로써 방향성 정의 알고리즘이 올바르게 동작함을 알 수 있다. 그러나 예상과 다르게 6번 노드에서 송·수신 메시지의 양이 가장 높

게 나타났는데 이는 싱크 노드의 위치가 목표지역의 4 사분면 쪽에 위치해 있고 목표지역 내에서 6번 노드가 싱크 노드와 가장 근접해 있으므로 중간 노드에서 발생하는 메시지를 중복해서 송·수신하여 메시지의 비율이 높게 나타난 것으로 예상된다. 또한 0번·8번 노드에서 메시지의 발생 비율이 6번 노드에 비해 상당히 작게 나타났으며 두 노드에서의 비율이 거의 비슷하게 나타났다. 본 논문에서 제안된 라우팅 프로토콜은 GEAR에서처럼 목표지역 안과 밖에서 사용되는 알고리즘을 달리 하지 않았다. 때문에 목표지역 내에 위치한 0·6·8번 노드에서 발생하는 메시지의 비율은 세 노드가 비슷하게 나타나야 하며 6번 노드의 경우와 마찬가지로 방향성의 영향을 받는다면 6번 8번 0번의 순으로 발생하는 메시지의 비율이 약간씩 줄어야 하지만 결과는 그렇게 나오지 않았다. 이러한 결과에 대한 원인은 현 시점에서는 정확히 알 수 없고 좀 더 많은 실험을 통하여 분석하여야 할 것으로 생각되므로 향후 연구 과제로 남겨둔다.

4.2 제안된 프로토콜과 GEAR의 성능 비교

본 논문에서 제안된 프로토콜의 성능을 GEAR와 비교하기 전에 GEAR가 Directed Diffusion보다 더 좋은 성능을 낼 수 있는지 확인해 보았다. 네트워크의 크기는 가로·세로 100m, 목표지역은 가로·세로 20m로 네트워크를 형성한 지역의 정 중앙에 일정하게 고정시켰다. 노드의 수는 10개부터 50개까지 10개씩 증가시켰고 노드의 수를 늘릴 때마다 10번씩 시뮬레이션을 수행하여 총 50번의 시뮬레이션을 거쳤다. 시뮬레이션 시간은 매회 500초로 하고 Directed Diffusion과 GEAR를 라우팅 프로토콜로 하였을 때, 발생하는 메시지의 양을 측정하여 평균을 내었다. 각 노드에서 소모되는 에너지의

양은 통신할 때 가장 많이 소모되므로 발생한 송·수신 메시지의 양은 네트워크에서 소모되는 에너지를 가늠할 수 있는 측정 자료가 될 수 있다. 시뮬레이션 결과는 <그림 4-3>과 같다.



<그림 4-3> Directed Diffusion과 GEAR의 비교
<Fig. 4-3> Directed Diffusion vs. GEAR

발생하는 메시지의 양은 노드의 수에 비례하여 증가한다. 노드의 수가 10개일 때 두 프로토콜에서 발생하는 메시지의 양은 거의 동일하게 나타났다. 하지만 노드의 수가 늘어남에 따라 두 프로토콜에서 발생하는 메시지 양의 증가율이 점점 커지는 것이 관찰 된다. 노드가 50개일 때 Directed Diffusion에서 발생하는 메시지의 양은 기하급수적으로 증가하여 거의 10만에 달하는데 반해 GEAR에서는 증가율이 그리 많지 않으며 노드가 50개일 때 발생하는 메시지의 양은 2만을 조금 웃도는 수준으로 Directed Diffusion의 24% 정도에 불과하다.

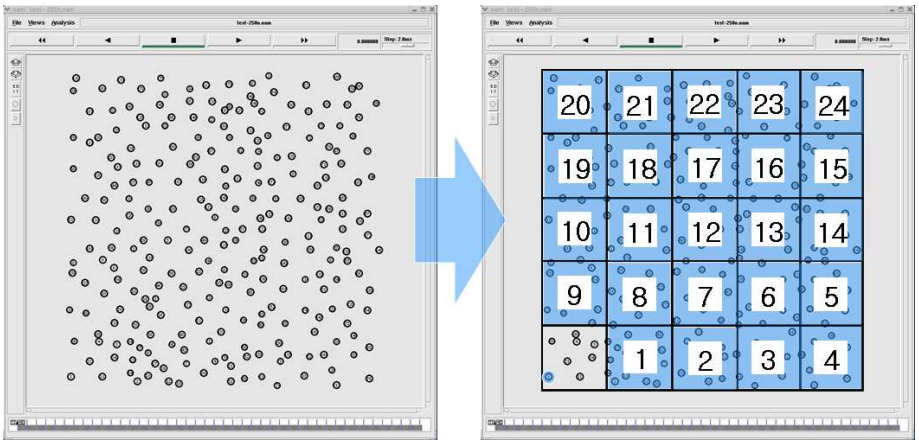
시뮬레이션에 사용된 최대 노드의 수가 50개로써 WSN에서는 적은 양임에도 불구하고 발생한 메시지의 양에서 두 프로토콜이 이렇듯 큰 차이를 보이는 이유는 메시지의 전달 방식 때문이다. 2장에서 분석했듯이 Directed Diffusion은 reinforcement 경로가 설정되기 전까지 interest와 event 메시지를 전송할 때 브로드캐스트를 사용한다. 이에 비해 GEAR는 이웃 노드들의 위치좌표와 잔류에너지 레벨을 이용해 다음 홉을 선택하므로 발생하는 메시지의 양을 효율적으로 줄인다. 프로토콜의 코딩 스타일이나 시뮬레이션의 환경에 따라 결과에 다소 차이는 있을 수 있으나 GEAR가 Directed Diffusion에 비해 효율적이라는 사실에는 변함이 없을 것이다.

GEAR가 Directed Diffusion보다 메시지 발생 양을 줄일 수 있음을 확인하였으므로 본 논문에서 제안된 프로토콜과 GEAR를 비교해 보고자 한다. 시뮬레이션을 위한 시나리오는 <표 4-1>을 바탕으로 작성하였다.

<표 4-1> 시나리오 설정
<Table 4-1> Scenario Configuration

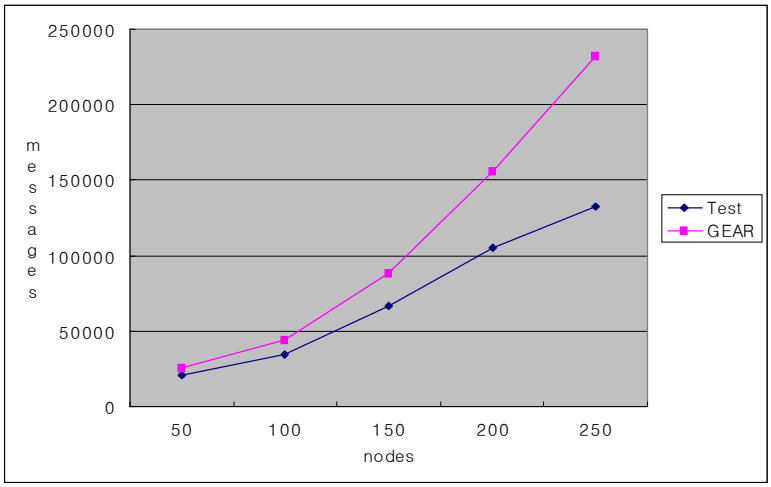
채널	Wireless Channel
신호확산모델	Two Ray Ground
물리계층	IEEE 802.11
안테나	Omni-directional Antenna
네트워크 크기	100m x 100m
노드수	50, 100, 150, 200, 250
시뮬레이션 시간	500s
노드의 이동	Random Function

각 노드의 위치는 NS-2에서 제공하는 랜덤함수를 이용해 임의로 위치시켰고 노드의 위치 또한 임의적으로 변화하게 하였다. 노드의 수는 50개부터 250개까지 50개씩 늘렸다. 시뮬레이션 과정에서 중요한 고려사항은 목표지역을 어디로 위치시키느냐이다. 목표지역의 위치에 따라 발생하는 메시지의 양에 많은 차이를 보이기 때문이다. 본 논문에서는 <그림 4-4>와 같이 전체 네트워크 구역을 25등분하여 싱크 노드가 위치한 하위구역을 제외한 1번부터 24번까지의 하위구역을 목표지역으로 정하고 노드의 수를 50개씩 늘릴 때마다 24번씩 시뮬레이션을 수행하였다. 다시 말하면, 노드의 수가 50개 일 때, 목표지역을 1번 구역부터 24번 구역까지 번호순서대로 바꾸면서 총 24번의 시뮬레이션을 거쳤다. 노드의 수가 100개일 때도 마찬가지 방법으로 24번의 시뮬레이션을 거쳤다. 이런 식으로 노드의 수를 50개씩 늘릴 때마다 같은 방법으로 시뮬레이션을 거쳐 총 120번의 시뮬레이션을 수행하였다. 시뮬레이션은 매회 500초 동안 수행하였고 각 노드에서 발생한 송·수신 메시지를 측정하여 평균치를 구하였다. 시뮬레이션 결과는 <그림 4-5>와 같다.



<그림 4-4> 시뮬레이션 과정

<Fig. 4-4> Simulation Process



<그림 4-5> 시뮬레이션 결과
<Fig. 4-5> Simulation Result

GEAR의 경우, 노드의 수가 150개 이상이 되면서 발생 메시지가 노드의 수와 정비례에 가까운 선형적 증가율을 보이고 있다. 이에 비해 제안한 프로토콜은 메시지 양의 증가율이 상대적으로 낮으며 노드의 수가 250개 일 때는 GEAR에 비해 43%정도 낮은 양을 보이고 있다.

이러한 결과는 GEAR가 설계단계에서 노드의 이동성을 고려하지 않은 것이 가장 큰 원인이며 또한 목표지역 내에서 반복 지리적 전달과 제한적 플러딩을 사용하기 때문에 목표지역 내에서 비약적으로 메시지의 양이 증가하였기 때문인 것으로 분석된다. 이에 비해 제안된 프로토콜은 방향성 설정 단계와 다음 홉 선정 단계를 통해 노드의 이동성을 지원하도록 설계되었으며 목적지의 방향성을 바탕으로 메시지를 발생하고 방향성에서 벗어난 노드의 메시지 발생을 억제함으로써 메시지 발생 수를 상대적으로 감소시킬 수 있었다.

제 5 장 결 론

본 논문에서는 WSN을 위해 제안된 기존의 라우팅 프로토콜 중 Directed Diffusion과 GEAR를 분석하여 두 알고리즘의 취약점을 도출하였고 각 노드의 위치좌표를 이용하여 네트워크에서 발생하는 메시지를 줄일 수 있는 라우팅 프로토콜을 제안하였다. 각 노드는 배포 직후 목적지로의 방향성을 설정하고 방향성에 입각한 자신의 이웃 노드 중 목적지에 가장 근접한 노드를 다음 홉으로 결정한다. 메시지를 수신한 노드들은 자신의 위치가 목적지까지의 방향성에 입각하는지를 판단하고 결과가 긍정이면 메시지를 전달, 부정이면 폐기함으로서 메시지 발생 횟수를 제어한다. 또한 노드가 이동하였을 경우 자신의 이동을 이전 이웃 노드들에게 알림으로써 방향성 설정과 다음 홉 선정을 다시 수행하도록 하여 노드의 이동성을 지원하도록 하였다. 시뮬레이션을 통하여 방향성 정의 알고리즘의 정확성을 입증하였고 GEAR와 비교하여 보다 효율적인 성능을 검증하였다.

그러나 시뮬레이션 수준에서, 발생하는 메시지의 수만을 측정하였기 때문에 GEAR와의 비교가 정확하게 이뤄졌다고 단정할 수는 없다. 이는 GEAR가 노드의 이동성을 지원하지 않도록 설계되었기에 더욱 그러하다. 본 논문에서 제안한 프로토콜이 기존의 프로토콜들보다 에너지 효율적인 프로토콜이라는 것을 입증하기 위해서는 실제 센서 노드에서 프로토콜을 동작시켜 네트워크의 수명을 측정해 보아야 할 것이다. 뿐만 아니라, Directed Diffusion에서 사용한 데이터 네이밍 모델과 데이터 병합 기법을 본 프로토콜에 그대로 적용하였으므로 프로토콜의 더 나은 성능을 위해서는 이 부분 역시 보완하여야 할 것이다.

참고문헌

[1] 정보통신부 홈페이지, http://www.mic.go.kr/index.jsp

[2] 김희철, 홍원기, 이종혁, 김현철, "USN 기반 환경정보 검색 시스템 선도연구", 한국인터넷진흥원, 2005.

[3] J. M. Kahn, R. H. Katz, and K.S.J. Pister "Next Century Challenges: Mobile Networking for "Smart Dust"", Proc. of the 5th annual ACM/IEEE international conference on Mobile computing and networking, Seattle, WA, pp.271-278, 1999.

[4] Berkeley Sensor and Actuator Center, http://www-bsac.eecs.berkeley.edu/

[5] K. Akkaya, M. Younis, "A Survey on Routing Protocols for Wireless Sensor Networks", Department of Computer Science and Electrical Engineering University of Maryland, Baltimore County Baltimore, MD 21250, 2003.

[6] Mesh Network Equipment - Municipal, WiFi, Broadband, Wireless, Mobile | Sensoria, http://www.sensoria.com

[7] UCLA Computer Science Department, http://www.cs.ucla.edu

[8] Mobicom 2002 Tutorial "Wireless Sensor Networks", http://nesl.ee.ucla.edu/tutorials/mobicom02/slides/Mobicom-Tutorial-2-MS.pdf

[9] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion for Wireless Sensor Networking", IEEE/ACM Transactions on Networking, Vol.11, No.1, pp.2-16, 2003.

[10] Y. Yu, R. Govindan, and D. Estrin, "Geographical and Energy Aware Routing: a recursive data dissemination

protocol for wireless sensor networks", UCLA Computer Science Department Technical Report UCLA/CSD-TR-01-0023, 2001

[11] F. Silva, J. Heidemann, R. Govindan, "Network Routing Application Programmer's Interface (API) and Walk Through 9.0.1", http://www.isi.edu/scadds/publication.html, 2002.

[12] TinyOS Technology Exchange, http://www.eecs.berkeley.edu/~culler/tinyos/ttx/

[13] David Culler's homepage, http://www.cs.berkeley.edu/~culler/

[14] KETI(Korea Electronics Technology Institute), http://www.keti.re.kr/

[15] (주)맥스포, http://www.maxfor.co.kr

[16] T.Liu, C. M. Sadler, P. Zhang, M. Martonosi, "Implementing Software on Resource-Constrained Mobile Sensors: Experiences with Impala and ZebraNet", in Proc. ACM 2nd International Conference on Mobile Systems, Applications and Services, Boston, MA, USA, 2004.

[17] A. Papadopoulos, J. A. Mccann, "Towards the Design of an Energy-efficient, Location-aware Routing Protocol for Mobile, Ad-hoc Sensor Networks",in Proc. IEEE 15th International Workshop on Database and Expert Systems Applications, Zaragoza, Spain, 2004.

[18] W. Adejie-Winoto, E. Schwartz, H. Balakrisshnan, and J. Lilley, "The design and implementation of an intentional naming system", in Proc. ACM Symp. Operating Systems Principles, Charleston, SC, pp.186-201, 1999.

[19] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, "Building efficient wireless sensor networks with low-level naming", in Proc. ACM Symp. Operating Systems Principles, Banff, Canada, pp. 146-159, 2001.

[20] E. M. Royer, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", in Proc. IEEE Personal Communications, pp.46-55, 1999.

[21] K. Fall, K. Varadhan, "The ns Manual", A Collaboration bet ween researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC, 2006.

[22] AWK (programming language) - Wikipedia, the free encyclo pedia, http://en.widipedia.org/wiki/Awk