

자산관리 시스템을 위한 RSS 라우팅 알고리즘에 관한 연구

A Study on the RSS Routing Algorithm for Asset Management System

*이민구 **강정훈 **임호정 **유준재 **윤명현

Mingoo Lee, Junghoon Kang, Hojung Lim, Junjae Yoo, Myunghyun Yoon

Abstract - Even though a lot of routing algorithms have been proposed, an omnipotent algorithm of routing technique, which has optimal efficiency, does not exist.

Therefore, A routing algorithm in a sensor network is an application oriented: the best effective routing algorithm depends on which application it is used to.

In this paper, the routing algorithm is proposed for the purpose of monitoring a movement of Assets in office. This Paper proposes a new multi-hop routing algorithm, that is, RSS(Received Signal Strength) value which was used in a localization of sensor network is applied to routing algorithm.

Key Words : Sensor Network, RSS, Routing Protocol

1. 장 서 론

유비쿼터스(Ubiquitous) 컴퓨팅 사회의 도래를 예상하는 주장들이 사회 곳곳에서 연일 제기되고 있다. 불과 수년 만에 전혀 생소하던 유비쿼터스라는 용어가 이제는 현 사회에서 무척 익숙해진 느낌이 든다.

본 논문에서는 유비쿼터스 컴퓨팅의 핵심 기술인 센서 네트워크(Sensor Network)를 이용하여 자산관리 어플리케이션을 효율적(반응 속도, 리소스 사용량)으로 관리하기 위한 라우팅 알고리즘 기술 개발에 관한 제안이므로 먼저, 센서 네트워크에 대해서 간단히 살펴보기로 한다.

센서 네트워크 분야는 U.C 버클리의 Culler 교수팀이 이끄는 NEST(Network Embedded Systems Technology) 프로젝트에 의해 리딩되고 있다. 센서 네트워크는 단어 그대로 기존의 센싱 기능에 국한되어 사용된 센서들에게 무선 통신을 통한 네트워킹 기능을 부여해 준 기술로서 센서를 통해 획득한 환경 데이터를 적절히 처리한 후, 무선 RF를 통해 목적지까지 효율적으로 전달해 주자는 취지의 기술이다.

이와 같은 센서 네트워크 기술의 데이터 전송 효율성을 높이기 위한 노력들 가운데 하나가 라우팅 알고리즘의 개발이다. 즉 목적지까지 가장 신속하게 센싱 데이터를 보낼 수 있는 라우트를 효율적으로 결정하기 위한 새로운 방법을 찾는 것이다.

센서 네트워크를 위해 많은 라우팅 알고리즘들이 제안되었지만, 모든 상황에서 최상의 효과를 낼 수 있는 만능의 라우팅 알고리즘은 존재하지 않는다. 즉, 센서 네트워크 분야에서 라우팅 알고리즘은 어플리케이션 지향성이라고 말할 수

있다. 어떤 서비스 혹은 어떤 어플리케이션에 사용할 것인지에 따라 가장 효율적인 라우팅 알고리즘이 결정될 수 있기 때문이다.

그리고 한 가지 명심할 점은, 무선 센서 네트워크가 저전력의 RF를 통해 통신이 이루어지고, 한정된 공간에서 제한적인 리소스(메모리, 전원 등)로 동작해야만 하는 센서 노드들로 네트워크를 이루기 때문에 기존의 애드 혹 네트워크에서 사용한 라우팅과는 본질적으로 다르다는 것을 명심해야 한다.

본 논문은 사무실 내에서 자산들의 이동을 모니터링 하기 위한 자산관리(Asset Management) 어플리케이션에서 사용될 라우팅 알고리즘 개발을 목표로 하였다. 이를 위해 센서 네트워크의 위치인지에서 사용된 적이 있는 RSS(Received Signal Strength) 값을 라우팅 알고리즘에 적용하여 반응속도가 빠른 새로운 멀티-홉 라우팅 알고리즘을 제안하였다.

본 논문에서 제안한 RSS 값을 이용하는 멀티-홉 라우팅 알고리즘은 자산관리를 위한 어플리케이션에 있어서 반응속도가 빠르다고 알려진 DSDV 기반의 라우팅 알고리즘에 비해 라우트 완성 속도 측면에서 27%, 리소스(RAM) 사용 측면에서 53%의 개선효과가 있음을 실험을 통해 확인하였다.

2. 장 본 론

2.1 절 TIP 30C

센서 네트워크의 구성을 위한 센서 네트워크 플랫폼으로 가장 잘 알려진 것은 U.C 버클리의 MOTE이다. 이 같은 MOTE는 1999년 처음 발표된 이후 다양한 형태(DOT, MICA, MICA2 등)로 발전되어 오고 있다.

본 논문에서 실험에 사용한 센서 네트워크 플랫폼은 '전자부품연구원'에서 2004년 개발한 TIP 30C 이다. 그림.1처럼 TIP 30C는 915MHz 무선 RF 대역을 사용하는 Inverted

저자 소개

* 正 會 員 : 전자부품연구원 유비쿼터스 컴퓨팅 연구센터

** 正 會 員 : 전자부품연구원 유비쿼터스 컴퓨팅 연구센터

F-Type 세라믹 안테나를 채택하였고, 다양한 센서(빛, 온도, 습도 등)들의 기능을 지원하고 있다. 마이크로 컨트롤러는 Atmega 128을 사용하였으며, 프로그램을 다운로드 할 수 있는 128KB의 플래시 메모리와 4KB의 RAM을 사용한다. 전원은 AAA 사이즈의 1.5V 배터리2개를 직렬로 연결하여 3V 전원을 공급해주고 있다.



<그림.1> 사용한 센서 노드(TIP 30C)

TIP 30C 센서 네트워크 노드는 CC 1000 이라는 초저전력 RF 송·수신 모듈을 채택하고 있다. CC 1000 RF 모듈은 RF의 세기를 측정하여 RSSI(Received Signal Strength Indicator) 출력 포트로 해당 신호를 출력하여 이를 ATmega 128의 ADC 포트에서 입력받게 된다.

출력된 RSSI 신호는 ATmega 128 A/D Converter에서 0V ~ 1.2V 범위로 측정이 가능하며, 노드 간의 거리가 가까울수록 0V에 가까운 값이 측정된다.

ATmega 128은 VDD 3V에서 동작하고 아날로그 신호를 10비트를 사용하여 샘플링하기 때문에 신호의 세기를 읽은 값은 0~400의 범위를 가지게 된다. (본 논문에서 실험을 통해 측정된 수치는 200을 넘지 않는다)

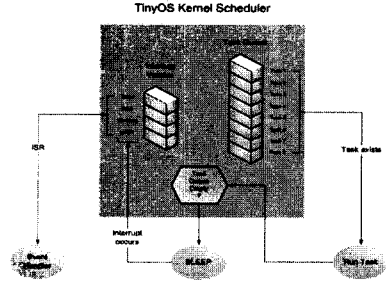
이상에서 살펴본바와 같이 수신된 신호의 세기인 RSS 값은 네트워크 상에서 링크의 성능을 나타내는 수치로 환산하여 라우팅 알고리즘의 구현에 이용할 수 있다. 본 논문에서는 이와 같이 센서 네트워크를 위하여 RSS 값을 이용한 새로운 개념의 라우팅 알고리즘을 구현하였다.

2.2 절 TinyOS 와 NesC

본 논문에서 센서 네트워크의 노드를 위한 OS는 UC 버클리에서 배포한 TinyOS를 사용하였다. TinyOS는 센서 네트워크와 같은 네트워크 임베디드 시스템(Network Embedded System)들을 위해 특별히 만들어진 OS이다. 이는 이벤트 기반의 어플리케이션, 소형의 코어 OS(400 바이트 정도의 코드), 작은 데이터 메모리를 갖는 초소형 용량의 OS를 만들기 위해 고안되었으며, 이벤트 기반의 단순 스케줄러 기능만을 제공한다.

TinyOS는 그림.2에서처럼 스케줄러의 기능만을 지원하는 제한적 기능의 OS라고 할 수 있다. TinyOS는 FIFO 구조의 태스크 큐(Queue)에 처리해야 할 태스크의 유무를 확인하고 처리해 준다. 이 과정이 끝나면, Sleep 상태에 빠지게 되고, 이벤트가 발생하게 되면 이벤트 핸들러가 인터럽트 서비스 루틴(ISR)에 따라 인터럽트 벡터 내에 있는 상용하는 동작을 취하게 된다. 이 과정을 완료하면 다시 Sleep 상태로 돌아가게 된다.

이처럼 센서 네트워크에서 사용하기 위해 고안되었던 TinyOS는 Sleep 상태로 유지하는 비율이 전체 시간에서 99% 이상으로 동작하여 장시간 노드의 지속 시간을 유지할 수 있도록 설계되어 있다.



<그림.2> TinyOS 커널 스케줄러

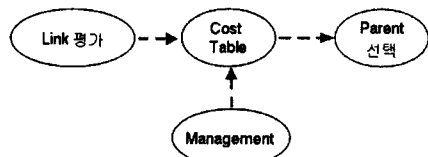
본 논문에서 프로그래밍에 사용한 언어는 NesC이다. NesC는 TinyOS의 특성들을 지원하기 위해 UC 버클리에서 만들어 배포한 프로그래밍 언어이다. 즉, NesC 어플리케이션들은 컴포넌트와 양방향의 인터페이스들로 구성되고, 태스크와 이벤트들에 기반을 둔 동시성 모델을 제공한다라는 점이 특징이다.

NesC는 커다란 관점에서 보면, C 언어의 확장 정도로 볼 수 있다. 이처럼 C 언어에 근거를 둔 이유는, C 언어가 마이크로 컨트롤러들을 위한 효율적인 코드 생성이 가능하고, 하드웨어에 액세스하기 위해 필요한 기본적인 특성들의 지원이 가능하며, 기존의 C 코드와의 상호작용이 단순하다는 장점 때문이다. 물론 기존의 많은 프로그래머들이 C 언어에 익숙해져 있다는 점도 NesC가 C 언어에 근거를 둔 큰 이유이다.

하지만, 위에서 언급한 C 언어의 장점만으로는 NesC의 초소형, 초저전력, 컴포넌트 기반의 구조라는 특성을 만족시키는 데에는 부족함이 있다. 즉, C 언어는 어플리케이션들을 구조화해야한다는 부분에서 만족스럽지 못하다. 결국 NesC는 컴포넌트라는 개념에 기반을 두어 TinyOS의 이벤트 기반의 동시성을 지원하고, 공유된 데이터의 동시 액세스가 가능하도록 지원하고 있다.

2.3 절 RSS를 이용한 라우팅 프로토콜

본 논문에서 무선 센서 네트워크를 이용한 자산관리 어플리케이션을 위해 RSS를 이용한 라우팅 알고리즘을 구현하였다. 이를 위해 본 논문에서 센서 노드들 간의 라우트 정보를 주고받아 목적지 노드에 도달하기 위해 네트워크 내의 주변 노드들 사이의 라우트를 결정하는 알고리즘의 동작 개요도는 그림.3과 같다.

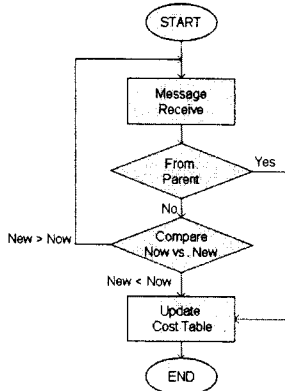


<그림.3> RSS를 이용한 라우팅 알고리즘 동작 개요도

본 논문에서 개발하고자 하는 라우팅 알고리즘의 최종 목표는 메시지를 전송하기 위한 목적지 노드까지 도달하기 위한 가장 성능이 좋은 라우트를 효율적으로 찾아내는 방법을 알아내는 것이다. 이를 위해 RSS를 이용한 라우팅 알고리즘은 그림.3과 같이 크게 4가지의 컴포넌트들로 구성되었다.

센서 네트워크를 위한 RSS를 이용한 라우팅 전체 동작에

대한 순서도를 그려보면 그림 4와 같다.



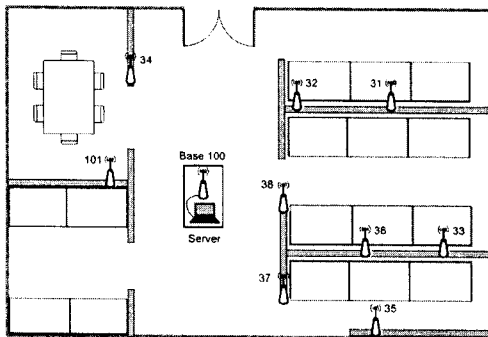
<그림 4> RSS를 이용한 라우팅 전체 동작 순서도

센서 노드들은 통신 네트워크를 유지하기 위해 라우트 메시지를 지속적으로 송·수신한 후, 라우트 메시지를 수신할 때마다 수신 신호의 세기인 RSS를 판단하여 센서 노드들 간의 링크의 품질을 계산한다. 센서 노드가 더 좋은 링크를 찾게 되면, Cost 테이블을 업데이트 하게 된다. 이후 다음부터는 메시지를 포워드 할 때에, 목적지의 어드레스에 도달하기 위한 새로운 Parent 노드를 설정하고, 이 센서 노드에 데이터를 포워드하게 된다.

2.4 절 라우팅 알고리즘 적용 및 실험 환경

그림 5는 두 가지 라우팅 알고리즘의 성능을 비교하기 위한 실험 환경이다. 센서 노드는 베이스 노드를 포함하여 총 10개의 TIP 30C가 준비되었고, 센서 노드의 높이를 일치시키기 위해 사무실내의 칸막이 위에 센서 노드들을 배치하였다.

두 가지의 라우팅 알고리즘은 동일한 테스트 조건을 확보하기 위해 두 라우팅 알고리즘을 적용했을 경우 같은 네트워크 토폴로지를 형성하는 위치에 센서 노드들의 위치를 고정시키고 측정하였으며, 각각의 라우팅 알고리즘의 반응 속도를 측정할 때 새로운 배터리로 교체하여 사용하였다. 실험상의 조작 오차로 인한 에러율을 줄이기 위해 동일한 조건에서 각각 100회 측정을 반복하여 얻은 값을 평균값 취한 후 결과값으로 사용하였다.



<그림 5> 라우팅 알고리즘 반응 속도 실험환경

2.5 절 실험 결과

표 1은 본 논문에서 센서 네트워크를 위해 구현된 DSDV

기반의 라우팅 알고리즘과 RSS 값을 이용한 라우팅 알고리즘간의 성능을 비교 분석하여 요약한 테이블이다. 반응 속도의 결과는 센서 노드가 9개인 경우를 기준으로 측정된 값이다.

| 항목 | DSDV 기반 라우팅 | RSS 이용 라우팅 | 개선율(%) |
|---------|-------------|------------|--------|
| 반응 속도 | 47 (sec) | 34 (sec) | 27 |
| RAM 사용량 | 1928 (Byte) | 912 (Byte) | 53 |

<표 1> 라우팅 알고리즘 성능 비교

실험 결과를 통해 알 수 있듯이, 본 논문에서 제안한 라우팅 알고리즘은 반응 속도 측면에서, 노드의 수가 9개인 경우, RSS를 이용한 라우팅 알고리즘은 DSDV 기반의 라우팅 알고리즘에 비해 네트워크 토폴로지를 완성하는 데 소요되는 시간을 약 27% 줄이는 효과를 거두었다. 그리고 RSS 값을 이용한 라우팅 알고리즘의 RAM 사용량의 크기는 0.912K 바이트로서 DSDV 기반의 라우팅 알고리즘에서의 RAM 사용량의 크기인 1.928K 바이트에 비해 약 53%의 절감 효과를 거두고 있음을 확인할 수 있다.

3. 장 결론

본 논문은 자산관리 어플리케이션이 요구한 빠른 반응 속도라는 측면과 일반 센서 네트워크 기술이 요구하는 리소스 사용량을 줄여야 한다는 두 가지 요구사항을 충족시킬 수 있는 라우팅 알고리즘을 구현하기 위해 RSS 값을 이용한 라우팅 방법을 제안하였다.

본 논문의 실험환경에서 RSS 값을 이용한 라우팅 알고리즘은 DSDV 기반의 라우팅 알고리즘에 비해 반응속도 측면과 리소스 사용 측면에서 우수함을 입증하였다. 하지만 다른 환경에서도 자산관리를 위해 본 논문이 제안한 라우팅 알고리즘을 그대로 적용할 경우 그 효능에 대해 단언하기 어렵다. 이는 비단 본 논문에서 제안하는 라우팅 알고리즘에만 해당되는 것이 아니다.

그러므로 센서 네트워크를 위한 라우팅 알고리즘은 적용되는 어플리케이션의 요구사항과 적용하고자 하는 환경에 맞게 최적화하는 작업이 반드시 필요할 것이다.

참 고 문 헌

- [1] David E. Culler, Jason Hill, Philip Buonadonna, Robert Szewczyk, and Alec Woo. "A Network-Centric Approach to Embedded Software for Tiny Devices" EMSOFT 2001, Oct 2001.
- [2] Joseph Polastre, Robert Szewczyk, Cory Sharp, David Culler. "The Mote Revolution: Low Power Wireless Sensor Network Devices" Proceedings of Hot Chips 16: A Symposium on High Performance Chips. August 22-24, 2004.
- [3] "ChipCon CC1000 Data Sheet," <http://www.chipcon.com/ies/CC1000 Data Sheet 2 1.pdf>