대규모 무선 센서 네트워크를 위한 확장성과 강건성이 있는 데이터 전송 방안

정회원 박 수 창*, 이 의 신*, 준회원 박 호 성*, 이 정 철*, 오 승 민*, 정 주 현*,
종신회원 김 상 하**°

Scalable and Robust Data Dissemination Scheme for Large-Scale Wireless Sensor Networks

Soochang Park*, Euisin Lee* ***Regular Members***, Hosung Park*, Jeongcheol Lee*,
Seungmin Oh*, Juhyun Jung* ***Associate Members***, Sang-Ha Kim**° ***Lifelong Member***

요 약

무선 센서 네트워크에서 데이터 전송은 데이터 중심 라우팅에 기반하여 이루어지기 때문에 공표/신청 통신 패러다임과 부합한다. 공표/신청 패러다임은 공간 분리성, 시간 분리성, 동기화 분리성이라는 세가지 분리 특성을 통해 대규모 애플리케이션 환경을 위한 확장성과 강건성을 제공할 수 있다. 그러나 현존하는 무선 센서 네트워크의 데이터 전송 방안들은 이 분리성들을 완전히 만족하지 못한다. 따라서, 우리는 세가지 분리성을 온전히 만족하기 위한 새로운 데이터 전송 방안인 ARBITER를 제시한다. ARBITER는 독립 네트워크 구조체를 구성하여, 공표자와 신청자 간의 정보 교환이 구조체를 통해 간접적이고 비동기적으로 이루어지도록 한다. ARBITER는 또한 공표자와 신청자가 서로 다른 시기에 연결을 시도하더라도 이를 지원할 수 있도록 구조체가 데이터와 쿼리를 저장하고 서로간의 매핑을 관리한다. 시뮬레이션 결과는 ARBITER가 확장성, 네트워크 강건성, 데이터 신뢰성, 이동성 지원, 그리고 에너지 효율성에서 더 나은 성능을 보인다는 것을 입증한다.

Key Words : Large-Scale Wireless Sensor Networks, Publish/Subscribe Paradigm, Mobile Sinks, Decoupling, Scalability, Robustness

ABSTRACT

In wireless sensor networks, data dissemination is based on data-centric routing that well matches the publish/subscribe communication paradigm. The publish/subscribe paradigm requires decoupling properties: space, time, and synchronization decoupling. For large-scale applications, the three decoupling properties provide scalability and robust communication. However, existing data dissemination schemes for wireless sensor networks do not achieve full decoupling. Therefore, we propose a novel data dissemination scheme that fully accomplishes the three decoupling, called ARBIETER. ARBITER constructs an independent network structure as a logical software bus. Information interworking between publishers and subscribers is indirectly and asynchronously performed via the network structure. ARBITER also manages storage and mapping of queries and data on the structure because of supporting different time connection of publishers and subscribers. Our simulation proves ARBITER show better performance in terms of scalability, network robustness, data responsibility, mobility support, and energy efficiency.

* 충남대학교 컴퓨터공학과 컴퓨터 네트워크 연구실 ({winter, hspark, eslee, badamul}@cclab.cnu.ac.kr)
** 충남대학교 컴퓨터공학과 컴퓨터 네트워크 연구실 (shkim@cnu.ac.kr) (°:교신저자)
논문번호 : KICS2009-07-293, 접수일자 : 2009년 7월 15일, 최종논문접수일자 : 2009년 11월월 16일

Ⅰ. 서 론

전형적인 무선 센서 네트워크는 싱크들과, 이벤트들, 그리고 많은 수의 센서 노드들로 구성된다[1]. 센서 노드들은 적은 비용으로 만들어지고, 적은 전력으로 동작하지만, 다양한 기능을 가진 장치이다. 대규모 센서 네트워크를 구성하기 위해, 무수히 많은 작은 센서들이 광활한 지역에 무작위로 분포된다[1],[2]. 무선 센서 네트워크에서의 데이터 전송은 전통적인 네트워크의 주소 중심 라우팅과는 다르게 데이터 중심 라우팅을 기반으로 한다[1],[5].

데이터 중심 라우팅은 주소가 아닌 데이터의 내용에 따라 라우팅이 이루어지는 공표/신청 통신 패러다임 (publish/subscribe communication paradigm)과 부합한다[2]. 공표/신청 패러다임은 관심이 증가되고 있는 추세이며, 대규모 애플리케이션의 분산된 정보 상호 작용에 적용하기 적절한 개념이다[4]. 이 패러다임에서 공표자와 신청자는 비동기적으로 이벤트 서비스라 불리는 논리적인 채널에 접속한다. 신청자는 관심 정보를 이벤트 서비스에 등록하며, 생산자에 의해 생성된 결과는 이벤트 서비스로의 공표를 통해 공개적으로 이용 가능해진다. 다시 말해, 이벤트 서비스는 관심 정보를 위한 논리적 상호작용이라는 추상적 개념으로서 소프트웨어 버스라 할 수 있다. 그림 1은 공표/신청 패러다임의 상호작용과 이를 무선 센서 네트워크에 적용했을 경우의 사례를 보여준다. 왼쪽 그림에서 다른 종류의 화살표는 서로 다른 유형의 공표와 신청을 나타낸다[2]. 오른쪽 그림은 싱크의 쿼리(query)에 의한 신청과 소스 노드의 데이터 공표를 보여준다.

공표/신청 통신 패러다임은 이벤트 서비스를 통해 공표자와 신청자 사이의 모든 명시적인 종속 상태를 제거함으로써 대규모 시스템을 위한 확장성과 통신 강건성을 제공한다. 게다가, 정보에 대한 공표와 신청의 분리는 서로 다른 개체들 사이의 관리와 동기화를 확실히 줄이며, 만들어진 통신 인프라가 이동 환경과 같이 선천적으로 비동기적인 분산 환경에 적절하도록 만든다[5]. 이벤트 서비스가 제공하는 공표자와 신청자 사이의 분리성(decoupling)은 공간, 시간, 동기화 이렇게 세가지 요소로 분해할 수 있다.

첫째로, 공간 분리성은 공표자들과 신청자들이 서로에 대해 알고 있을 필요가 없다는 것을 뜻한다. 다시 말해, 공표자들과 신청자들은 얼마나 많은 신청자들이 혹은 얼마나 많은 공표자들이 상호작용에 참여하는지 알 필요가 없다. 둘째로, 시간 분리성은 공표와 신청이 서로 다른 시간에 이루어질 수 있다는 것을 말한다. 공표자는 신청자가 연결되어 있지 않는 동안에도 어떤 이벤트를 공표할 수 있으며, 반대로 신청자는 해당 이벤트의 공표자가 연결되어 있지 않는 동안에도 정보를 얻을 수 있다. 마지막으로, 동기화 분리성은 주요 작업이 저지 되는 일 없이 비동기적으로 정보의 상호작용이 일어날 수 있다는 뜻이다. 다시 말해, 정보의 생성과 소비를 위해 공표자와 신청자가 직접적으로 그리고 지속적으로 제어 시그널링을 발생하지 않음으로써 비지 웨이팅(busy waiting) 상태에 빠지지 않도록 한다. 결과적으로 세가지 분리성을 만족함으로써, 공표자와 신청자 사이의 모든 종속성을 제거할 수 있다.

그러나 무선 센서 네트워크를 위한 기존의 데이터 전송 방안들[6]-[10]은 완전한 분리성을 이루지 못했다. 방안들[6]-[8]은 각각 소스 노드와 싱크로 알려

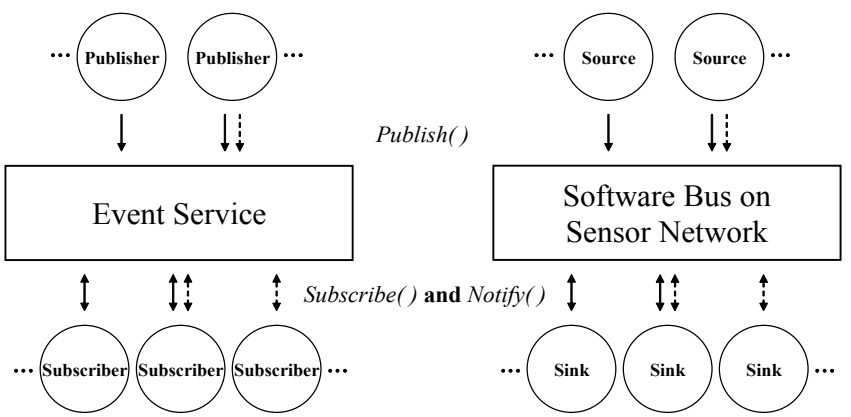


그림 1. 정보의 상호작용

진 공표자 혹은 신청자에 종속된 네트워크 구조체(structure)를 생성한다. 공간 종속성은 소스 노드나 싱크의 수에 비례하여 너무 많은 네트워크 구조체가 생성되는 결과를 낳는다. 따라서 공간 연결성은 확장성을 감소시킨다. 비록 다른 방안들[9],[10]은 앞서 언급한 방안들과 달리 독립된 네트워크 구조체를 사용한다고 해도, 소스 노드들과 싱크들이 정보 상호작용을 위해 같은 시간에 구조체에 참여해야 한다. 게다가, 거의 대부분의 전송 방안들에서 소스 노드들과 싱크들뿐만 아니라 네트워크 구조체를 구성하는데 참여하지 않는 많은 센서 노드들까지 동시에 데이터 공표와 신청을 위해 제어 시그널링에 참여해야 한다. 시간 종속성과 동기화 종속성은 이동성과 같이 네트워크에 동적인 요소가 발생했을 때, 강건성과 데이터 신뢰도에 심각한 영향을 미친다. 이는 빈번하게 네트워크 토폴로지 또는 데이터 전달 경로를 재설정하기 위한 제어 복잡도와 시그널링 부하의 증가 그리고 그로 인한 데이터 손실률 증가 때문이다.

이 논문에서 우리는 세가지 분리성을 온전히 만족하기 위한 새로운 데이터 전송 방안인 ARBITER (Architecture and data-centric Routing Based on and Independent sofTware bus over a wireless sEnsor netowRk)를 제안한다. ARBITER는 공간 분리성을 위해 논리적인 소프트웨어 버스인 독립 네트워크 구조체를 구성하며, 시간 분리성을 위해 구조체 상에서 데이터와 쿼리를 저장하고 서로간의 매핑(mapping)을 관리한다. ARBITER에서는 동기화 분리성을 보장하기 위해서, 소스 노드들과 싱크들뿐만 아니라 네트워크 구조체를 구성하는데 참여하지 않는 많은 센서 노드들까지 데이터 공표와 신청을 위한 제어 시그널링에 참여하지 않는다. 우리는 ARBITER 설계에서 무선 센서 네트워크에서 가장 중요한 이슈인 에너지 효율성을 고려했다. 또한 시뮬레이션을 통해 ARBITER가 다른 데이터 전송 방안들에 비해 확장성, 강건성, 신뢰성, 이동성 지원, 그리고 에너지 효율성의 측면에서 더 나은 성능을 보인다는 것을 입증한다.

Ⅱ. ARBITER 설계

이 단원에서는 기본 설계 원칙과 시스템 가정, 그리고 세가지 분리성을 ARBITER 설계에 적용하기 위한 함수와 메시지 정의를 설명한다.

2.1 설계 원칙

공표/신청 통신 패러다임은 공표자와 신청자 사이의 분리성을 통해 대규모 애플리케이션에서도 혹은 이동성과 같은 동적 환경에서도 확장성과 강건성을 증가시켜준다. 이 분리성은 공간, 시간, 동기화의 분리를 통해서 완전히 만족된다. 우리는 ARBITER가 온전한 분리성을 만족하게 하기 위해서, 각 분리 특성에 맞는 설계 원칙을 정의한다.

2.1.1 공간 분리성의 적용

공간 분리성은 공표자와 신청자가 데이터를 상호 교환하는 동안 서로에 대해 인지하고 있을 필요가 없다는 것을 의미한다. 다시 말해, 공표자와 신청자 사이의 통신이 독립 공용 구조체를 통해 간접적으로 이루어진다. 공표자는 구조체에게 데이터를 전송하고, 신청자 또한 구조체에게 데이터를 요청하며, 구조체는 데이터를 신청자에게 전달해준다. 따라서, 공간 분리성을 만족하기 위한 첫 번째 원칙으로 소스 노드들과 싱크들을 분리시키는 *독립 공용 구조체 구성*을 정의한다.

2.1.2 시간 분리성의 적용

시간 분리성은 데이터의 공표와 신청이 서로 다른 시간에 이루어질 수 있다는 것을 의미한다. 다시 말해서 공표자는 신청자가 연결되어 있지 않는 동안에도 어떤 이벤트를 공표할 수 있으며, 반대로 신청자는 해당 이벤트의 공표자가 연결되어 있지 않는 동안에도 정보를 얻을 수 있다. 이는 데이터와 쿼리가 네트워크 상의 중간 저장소에 보존되며, 그곳에 이들의 매핑 메커니즘이 제공된다는 것을 의미한다. 따라서 두 번째 설계 원칙으로 *데이터와 쿼리의 저장 및 매핑 지원*을 정의한다.

2.1.3 동기화 분리성의 적용

동기화 분리성은 공표자가 데이터를 생산하는 동안 차단되지 않는다는 것과 신청자가 다른 동시 작업을 수행하는 동안 공표된 데이터를 비동기적으로 얻을 수 있다는 특징을 제공한다. 다시 말해, 이는 공표자와 신청자가 상호작용을 위해 직접적으로 그리고 지속적으로 제어 시그널링에 참여하지 않음으로써 다른 작업을 차단하는 비지 웨이팅 상태에 빠지지 않는다는 것을 의미한다. 무선 센서 네트워크의 경우에 소스 노드는 이벤트가 발생해서 데이터가 생성된다면 단순히 공용 구조체에 데이터를 전송하며, 싱크는 지속적인 제어 시그널링 없이 구조

체에 데이터를 요청한다. 오직 구조체만 소스 노드로부터 수신한 데이터를 싱크에게 전달하기 위한 관리가 필요하다. 따라서 마지막 설계 원칙으로 *단순 공표, 단순 신청, 제어된 통지*를 고려한다.

2.2 시스템 가정

여기서는 제안 방안을 위한 가정들을 언급한다.

- 많은 수의 동일한 센서 노드들이 광활한 지역에 고르고 조밀하게 뿌려진 후 스스로 무선 센서 네트워크를 구성한다. 장거리 데이터 전달은 다중 홉 (multi-hop) 통신 방식을 통해 이루어진다.
- 모든 센서 노드는 전개 후에 위치 측정 기법[11] 혹은 GPS (Global Positioning System) 신호[12] 수신을 통해 자신의 위치를 알 수 있다.
- 다중 싱크와 다중 이벤트가 무선 센서 네트워크 상에서 이동한다. 모든 센서 노드가 쿼리를 위해서 이동 싱크의 에이전트(agent)로 동작할 수 있으며, 이동 이벤트에 대한 데이터를 생성하고 보고하기 위한 소스 노드로 동작할 수 있다.
- 센서 노드들은 목표 지역에 특정한 목적을 가지고 전개되기 때문에, 각각의 싱크는 비슷한 정보에 관심이 있다. 또한 센서 노드들은 몇 가지의 센싱 모듈(module)만을 탑재할 수 있기 때문에, 센서 노드들로부터 생성된 데이터는 몇가지 범주로 분류할 수 있다. 따라서 센서 노드들은 [13]에서 제안된 데이터 중심 저장 방안처럼 해시(hash) 함수를 이용해 각각의 데이터 타입에 따라 해당 저장 노드를 선택할 수 있다.

2.3 함수와 메시지 형식

이 소단원에서는 ARBITER가 동작하기 위한 세 가지 핵심 함수와 메시지 형식을 정의한다. 이 함수와 메시지들은 설계 원칙을 만족하기 위한 프로세스에 이용된다.

2.3.1 함수

그림 1에서 볼 수 있듯이, 공표/신청 통신 패러다임의 이벤트 서비스는 데이터 상호작용을 위해 세 가지 핵심 함수를 제공한다. 우리는 ARBITER 운용을 위해 무선 센서 네트워크에 적합한 함수를 정의한다.

- Subscribe (query message, location): 이 함수는 싱크의 에이전트 노드가 관심 데이터를 요청하기 위해 사용한다. Location은 에이전트 노드의 지리적 위치를 나타낸다. 쿼리 메시지는 위치 정보와 함께 위치 기반 포워딩 (geographic forwarding)[14] 방안을 사용하여 공용 구조체에 전달된다.
- Publish (data packet): 이 함수는 소스 노드가 생성한 데이터를 공용 구조체에 공표하기 위해 사용한다. 데이터 패킷 전달 또한 위치 기반 포워딩 방안을 사용한다.
- Notify (data packet): 이 함수는 공용 구조체에 저장된 데이터를 요청 싱크에게 전송하기 위해 사용한다. 사실상 데이터 패킷은 싱크의 에이전트 노드에게 전송된다. 데이터 통지는 위치 기반 포워딩과 멀티캐스팅을 사용하여 이루어진다.

2.3.2 메시지 형식

ARBITER 운용을 위해 세가지 메시지가 이용된다. 메시지들은 위치 기반 포워딩을 통해 전달되므로, 헤더(header)에 소스 위치와 목적지 위치가 들어간다. 메시지들의 형식은 다음과 같다.

- 초기화(initialization) 메시지 형식: 이 메시지는 무선 센서 네트워크에 독립 공용 구조체를 생성하고 관리하기 위해 사용된다. 이 메시지는 관리를 위한 정보를 포함한다. 관리에는 두 가지 모드가 있다. 첫 번째 모드는 초기화 노드, 저장 노드, 전달 노드와 같이 기준 위치에서 가까운 노드를 선출하기 위한 노드 선출 모드이다. 두 번째 모드는 초기화 노드가 저장 노드들과 전달 노드들의 위치 재배열, 즉 셀(cell) 범위 값 변경을 하기 위한 격자 재배치 모드이다.
- 쿼리 메시지 형식: 이 메시지는 싱크의 관심 데이터에 대한 명세, 싱크의 지역 타임스탬프 (local timestamp), 싱크의 식별자(identifier)를 포함한다. 쿼리 메시지는 싱크와 싱크의 에이전트가 사용한다.
- 데이터 패킷 형식: 이 패킷은 데이터의 페이로드 (payload)와 페이로드의 명세를 포함된다. 소스 노드의 데이터 공표와 저장 노드의 데이터 통지에 사용된다.

Ⅲ. ARBITER 운용

이 단원에서는 설계 원칙들을 만족시키며 에너지 효율성을 고려하기 위한 프로세스를 설명하고, 함수

들과 메시지들을 이용한 ARBITER의 기본적인 운용을 설명한다.

ARBITER는 설계 원칙들을 만족하기 위해 네 가지의 프로세스를 취한다. 첫째, 무선 센서 네트워크상에서 소스 노드와 싱크에 독립된 가상 격자(virtual grid)를 배치하고, 그 안에서 데이터 버퍼링을 위한 저장 노드를 정한다. 둘째, 가상 격자 상에서 데이터 융합(aggregation)과 멀티캐스팅을 지원하기 위한 전달 노드를 정한다. 셋째, 소스 노드는 publish() 함수만을, 싱크의 에이전트 노드는 subscribe() 함수만을, 저장 노드와 전달 노드는 추가적으로 notify() 함수를 사용한다. 넷째, 저장 노드들과 전달 노드들 주위의 핫 스팟을 피하기 위해 가상 격자를 재배치한다.

네 가지 프로세스를 기반으로 ARBITER 기본 운용은 다음과 같이 요약된다. 소스 노드는 이벤트가 발생했을 때 데이터 패킷을 생성하고, publish()를 통해 가장 가까운 전달 노드에게 패킷을 전송한다. 전달 노드는 데이터 패킷을 수신하면 가장 가까운 상위 레벨 전달 노드에게 전송한다. 이 과정은 데이터 패킷이 최상위 레벨 전달 노드에게 도착할 때가지 반복된다. 최상위 레벨 전달 노드는 [13]에서 제안된 데이터 중심 저장 방안처럼 해시 함수를 이용해서 데이터 타입에 따라 선택된 저장 노드에게 데이터 패킷을 전송한다. 데이터 패킷을 수신한 저장 노드는 데이터 패킷의 데이터 명세를 저장되어 있던 쿼리 메시지들의 명세와 비교한다. 만약 일치하는 쿼리 메시지가 없다면 저장 노드는 단순히 데이터 패킷의 데이터를 저장한다. 싱크는 원하는 데이터가 생길 경우 하나의 센서 노드를 에이전트 노드로 선택하고 쿼리 메시지를 전송한다. 에이전트 노드는 쿼리 메시지를 subscribe()를 통해 가장 가까운 전달 노드에게 전송한다. 이후 데이터 패킷과 같은 과정으로 정해진 저장 노드에게 전송된다. 쿼리 메시지를 수신한 저장 노드는 메시지의 명세를 저장되어 있던 데이터의 명세와 비교한다. 만약 일치하는 데이터가 없다면 저장 노드는 단순히 쿼리 메시지를 저장한다. 만약 데이터와 쿼리 메시지의 명세가 일치한다면 저장 노드는 해당 데이터로 데이터 패킷을 생성하고 notify()를 통해서 에이전트 노드에게 쿼리 경로의 역행 경로로 전송한다. 마지막으로 에이전트 노드는 싱크에게 데이터 패킷을 전송한다.

3.1 네트워크 구조체 구성

많은 수의 동일한 센서 노드들이 불규칙한 형태의 광활한 지역에 고르고 조밀하게 뿌려질 것이다. 센서 노드들은 목표 지점 (reference point)을 중심으로 뿌려지기 때문에 모든 노드들은 불규칙한 형태의 센싱 지역에서도 목표 지점의 위치를 알고 있다고 가정한다. 따라서, 목표 지점과 미리 정의된 셀 범위 값 α를 통해서 가상 격자를 생성할 수 있다. 모든 센서 노드들은 가상 격자 상에서 스스로 독립 공용 구조체의 구성을 시도한다. ARBITER는 우선 초기화 노드를 선정하며, 다음으로 저장 노드와 전달 노드가 분산적으로 정해진다. 그림 2에서 각 레벨의 각 교차 지점마다 저장 노드 혹은 전달

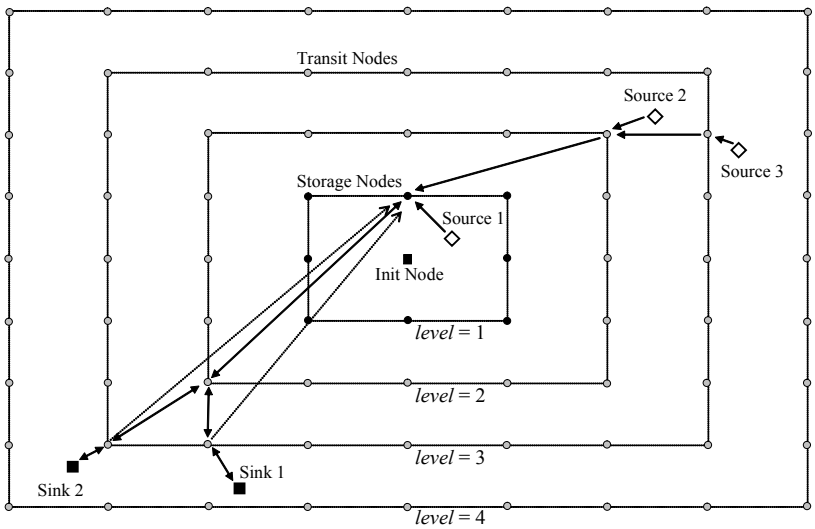


그림 2. 독립 공용 구조체를 통한 싱크와 소스 노드 사이의 정보 상호작용

노드가 배치된다. 레벨 1 은 저장 노드들만이 배치되며, 레벨 2 이상은 전달 노드들이 배치된다.

정확한 목표 지점의 위치에서 가장 가까운 노드를 초기화 노드로 선출하기 위해, ARBITER는 각 노드와 목표 지점 사이의 거리 'r'과 셀 범위 값 'α'의 반이라는 두 가지 요소를 고려한다. 목표 지점에서 α/2 범위 안의 모든 센서 노드들은 아래의 식(1)을 통해 대기 시간을 설정한다. 'a'는 대기 시간 차이를 확장하거나 줄이기 위한 상수이다.

$$\left\lfloor \left(r \div \frac{\alpha}{2} \times a \right) \right\rfloor \quad (1)$$

노드들은 설정된 시간 동안 대기한다. 이후 하나의 노드가 노드 선출 모드의 초기화 메시지를 생성하여 초기화 노드 후보 범위로 지역 플러딩 한다. 다시 말해 그 하나의 노드가 식(1)에 의해 가장 짧은 대기 시간이 설정된 목표 지점에 가장 가까운 노드이다. 센서 노드가 다른 노드로부터 초기화 메시지를 수신하면, 대기 설정을 취소하고 초기화 노드의 위치와 지역 고유 주소를 저장한다. 그림 3에서 회색 노드가 선출된 초기화 노드이다.

저장 노드들과 전달 노드들의 선출은 초기화 노드의 선출과 같은 방법으로 실행된다. 레벨 1 선상의 교차 지점들은 저장 노드들의 선출에 사용된다. 레벨 2 이상의 교차 지점들은 전달 노드들의 선출에 사용된다. 모든 선출은 분산적으로 실행된다. 하나의 노드 선출을 위한 지역 플러딩은 셀 범위 값 'α'로 이루어지기 때문에 독립 공용 구조체 생성은 한 번의 광역 플러딩과 같은 부하를 갖는다. 이는 광역 네트워크 구조체 구성이라는 면을 고려했을 때 과도한 통신 부하가 아니다.

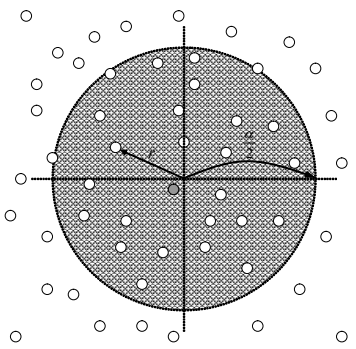


그림 3. 한 지점을 통한 실제 노드 배치 (초기화 노드, 저장 노드, 전달 노드)

ARBITER는 저장 노드들과 전달 노드들 주위의 핫 스팟 문제를 해결하기 위해 주기적으로 독립 구조체를 재생성 한다. 초기화 노드는 미리 정해진 시간이 지나면 새로운 목표 지점과 새로운 셀 범위 값을 광역 플러딩 하여 새로운 독립 구조체를 생성한다. 새로 선정된 초기화 노드는 일정 시간이 지나면 같은 역할을 반복한다. 초기화 노드는 정보의 상호 작용에 관여하지 않으며 오직 독립 구조체를 재생성 하는 역할만 담당한다.

셀 범위 값 'α'는 데이터의 융합과 멀티캐스트 효율성, 구조체 유지 비용 그리고 싱크와 소스 노드의 이동에 관련된다. α가 클수록 구조체 유지 비용이 적게 들며, 싱크와 소스 노드의 이동에 따른 업데이트 비용도 적게 든다. 그러나 데이터의 융합과 멀티캐스트의 효율성은 떨어진다. 반대로 작을수록 반대의 현상이 일어난다. 전달 노드의 개수는 그림 2에서 볼 수 있듯이 네트워크의 크기와 α값에 의해 정해진다. 하지만 저장 노드의 개수는 응용이 필요로 하는 정도에 따라 달라질 수 있다.

3.2 쿼리와 데이터의 저장과 매핑

선출된 저장 노드는 세가지 기능을 수행한다. 첫 번째 기능은 쿼리와 데이터의 버퍼링이다. 두 번째는 쿼리와 데이터가 일치하는지 비교하는 기능이다. 세 번째는 일치되는 데이터를 쿼리 경로의 역행 경로로 요청한 싱크에게 전송하는 기능이다.

저장 노드에서는 쿼리와 데이터를 정보의 중요성에 따라 다른 수명으로 버퍼링 한다. 또한 평균 온도와 같은 데이터는 융합하여 저장한다. 실제로 ARBITER에서는 데이터 요청을 위한 쿼리 메시지와 데이터의 공표와 통지를 위한 데이터 패킷이 제각기 사용된다.

쿼리 메시지는 싱크의 관심 데이터 명세, 싱크의 지역 타임 스탬프, 싱크의 식별자를 포함한다. 싱크의 관심 데이터 명세는 데이터의 명세와 비교하기 위해 쓰인다. 싱크의 지역 타임 스탬프는 싱크의 식별자를 통해 구별되는 같은 싱크가 보낸 쿼리 메시지들의 중복성을 확인하기 위해 쓰인다. 싱크의 쿼리를 보존하기 위해 저장 노드는 쿼리 메시지를 온전히 저장한다.

데이터 패킷은 데이터 페이로드와 페이로드의 명세를 포함한다. 데이터 페이로드는 실제 데이터이며, 페이로드의 명세는 쿼리 메시지의 명세와 같은 형식의 간단한 설명이다. 저장 노드는 명세를 통해 쿼리와 데이터가 같은 종류인지 비교할 수 있다. 저장

노드는 데이터 융합을 위해 데이터 페이로드만 저장한다. 따라서 융합된 데이터로 새로운 명세와 함께 요청을 위한 데이터 패킷을 생성한다. 저장 노드의 마지막 기능은 데이터의 통지이다. 다음 소단원에서 데이터의 통지에 대해 자세히 설명한다.

3.3 쿼리와 데이터 전송

싱크는 센싱 지역 상에서 한 이벤트의 정보에 관심이 생길 때 이웃 노드들 중 가장 가까운 노드를 에이전트 노드로 정하고, 에이전트 노드에게 쿼리 메시지를 전송한다. 에이전트 노드는 쿼리 메시지를 수신한 후, 격자 상에서 상위 레벨을 향해 가장 가까운 전달 노드에게 쿼리 메시지를 전송한다. 네트워크 구조체 초기화 과정에서 각 전달 노드는 자신의 존재에 대해 지역 플러딩 하기 때문에 에이전트 노드는 이미 상위 레벨 방향의 가장 가까운 전달 노드를 알고 있다. 간단하게 에이전트 노드를 생략한 그림 2에서 볼 수 있듯이, 에이전트 노드가 레벨 3에 혹은 레벨 4와 레벨 3 사이에 위치한다면 에이전트 노드는 레벨 3의 전달 노드들 중 가장 가까운 노드를 자신의 전달 노드로 선택한다. 에이전트 노드에서 선택된 전달 노드까지의 쿼리 메시지 전송은 Subscribe() 함수로 실행된다. 함수의 매개변수는 수신한 쿼리 메시지와 에이전트 노드의 위치이다. 쿼리 메시지는 위치 기반 포워딩 방안[14]을 통해서 선택된 전달 노드로 전송된다.

전달 노드는 쿼리 메시지를 수신하면, 상위 레벨의 전달 노드들 중에 자신과 미리 정의된 저장 노드를 연결하는 선상에 가장 가까운 노드를 다음 전달 노드로 선택한다. 전달 노드는 쿼리의 명세를 통해 쿼리 메시지의 해당 저장 노드를 알고 있기 때문에, 그림 2의 점선처럼 저장 노드까지의 목표 방향 선을 그릴 수 있다. 각 레벨의 전달 노드는 Subscribe()를 통해 상위 레벨의 전달 노드에게 쿼리 메시지를 전송한다. 쿼리 메시지가 레벨 2 전달 노드에게 도착하면, 레벨 2 전달 노드는 쿼리 메시지를 직접적으로 해당 저장 노드에게 전송한다. 만약 에이전트 노드가 레벨 2 사각형 안쪽에 위치한다면, 에이전트 노드는 곧바로 쿼리 메시지를 저장 노드에게 전송한다.

쿼리 메시지가 전달 될 때, 모든 에이전트, 전달, 저장 노드들은 모든 쿼리 메시지에 대해 이전 노드와 다음 노드의 위치 정보를 저장하고 관리해야 한다. 노드들이 모든 쿼리 메시지에 대해 정보 관리를 한다고 해도, 노드들은 단지 이전 노드와 다음 노드의 위치만을 저장하며, 전달 경로상의 모든 노드들이 아닌 에이전트, 전달, 저장 노드들만이 위치 정보 저장과 관리를 맡기 때문에 노드들의 버퍼링 부하는 높지 않을 것이다. 한 전달 노드가 둘 이상의 에이전트 노드 또는 전달 노드에게 같은 정보를 원하는 쿼리를 수신한다면, 그 전달 노드는 이전 노드들의 위치를 모두 저장하고 하나의 쿼리 메시지만 상위 전달 노드에게 전송한다. 이런 과정은 같은 쿼리의 중복 전달을 예방하고 데이터 상호 작용에서 에너지 소비를 줄인다.

소스 노드는 데이터 페이로드와 데이터의 명세로 데이터 패킷을 생성한다. 소스 노드 또한 Publish()를 통해 선택된 저장 노드에게 데이터 패킷을 전송하며, 쿼리 메시지의 전송과 같은 방법으로 실행된다. 그러나 소스 노드는 데이터 패킷의 전송에 대해 어떤 응답도 필요로 하지 않기 때문에, Publish()는 위치 정보를 포함하지 않으며 데이터 패킷의 전송 노드는 이전 노드의 위치를 관리하지 않는다.

저장 노드는 데이터와 쿼리 메시지의 매핑에 성공하면, 데이터 패킷을 생성한다. 저장 노드는 Notify() 함수를 이용하여 자신에게 쿼리 메시지를 전송한 에이전트 노드에게 이 패킷을 전송한다. 에이전트 노드는 싱크의 식별자를 통해 싱크에게 데이터 패킷을 전송한다. 그림 2는 ARBITER를 이용한 싱크들과 소스 노드들의 상호 작용을 보여준다.

데이터는 전달 노드들로 구성된 쿼리 메시지의 역 경로를 통해 저장 노드로부터 에이전트 노드까지 전송된다. 어떤 에이전트 노드들은 전달 노드들을 공유하므로, 데이터 전송은 그림 2에서 볼 수 있듯이 멀티캐스트 방법으로 실행될 수 있다.

3.4 이동성 관리

많은 보편적인 센서 애플리케이션에서 싱크는 센서 지역상에서 움직이며 관심 데이터를 요청한다. 싱크의 이동성을 지원하기 위해 ARBITER는 지역적인 이동에 궤적(trajectory) 라우팅을, 장거리 이동에는 전달 노드간 핸드오버(handover)를 사용한다. 동기화 분리성을 만족하기 위해서 ARBITER는 제어 시그널링에 싱크의 개입을 최소화한다.

궤적 라우팅은 소속된 전달 노드의 격자 셀 안에서 싱크의 지역 이동을 지원하기 위해 실행한다. 싱크는 에이전트 노드를 선택한 후에, 이웃 센서 노드들에게 일정한 간격으로 비콘 (beacon) 메시지를 브로드캐스트 한다. 비콘 메시지는 무선 링크 연결을 유지하기 위해 일반적으로 MAC 계층에서 필요로

하는 메시지다. 따라서 에이전트 노드는 싱크의 쿼리 메시지에 대한 데이터를 수신했을 때, 추가적인 제어 시그널링 없이 비콘 궤적을 따라 플러딩을 함으로써 데이터를 싱크에게 전송할 수 있다.

모든 싱크들은 주위 노드에게 목표 지점과 α값을 요청할 수 있기 때문에 전달 노드들과 저장 노드들을 포함한 독립 구조체의 구성을 알 수 있다. 싱크가 다른 격자 셀로 움직인다면, 싱크는 에이전트 노드를 다시 선택하고 새로운 전달 노드에게 쿼리 메시지를 전송한다. 쿼리 메시지의 재전송 과정 동안 그 경로상의 전달 노드가 같은 쿼리를 이미 수신한 적이 있다면, 그 전달 노드는 앞서 언급한 멀티캐스팅의 경우에서처럼 더 이상 쿼리 메시지를 전송하지 않고, 이전 노드의 위치만 관리한다.

새로운 에이전트 노드의 선정은 궤적의 역경로를 통해 이전 에이전트 노드에게 통보된다. 이전 에이전트 노드는 저장 노드로 향하는 경로를 삭제하기 위한 쿼리 메시지를 생성하여 자신의 전달 노드에게 전송한다. 이 쿼리 메시지는 싱크의 식별자를 포함하여 이전과 같은 명세를 가지고 있지만 타임스탬프 항목이 0으로 설정된다. 전달 노드는 타임스탬프 값이 0인 쿼리 메시지를 수신하면, 자신이 저장하고 있던 에이전트 노드의 위치와 쿼리 메시지를 삭제한다. 전달 노드는 또한 상위 전달 노드에게 쿼리 메시지를 전송하며, 상위 레벨의 전달 노드도 같은 과정을 수행한다. 타임 스탬프 값이 0인 쿼리 메시지에 의한 삭제는 같은 명세를 가진 다른 쿼리 메시지를 저장하고 있는 전달 노드를 만나거나 저장 노드를 만날 때까지 실행된다. 따라서 ARBITER는 추가적인 제어 시그널링 없이 단지 에이전트 노드를 재선출하는 것으로 싱크의 장거리 이동을 지원한다.

Ⅳ. 성능 분석

이 단원에서는 ARBITER의 성능을 평가하기 위한 시뮬레이션 환경과 평가 항목을 설명한다. 이 성능 분석을 수행하는 목표는 세가지 측면에 있다. 첫째는 ARBITER가 관련 연구인 TTDD[8]와 Directed Diffusion[6]보다 확장성의 측면에서 더 나은 성능을 제공할 수 있다는 것을 증명하는 것이다. 둘째는 ARBITER가 모든 싱크들과 모든 소스들 사이에서 높은 데이터 신뢰도를 제공한다는 것을 보이는 것이다. 마지막으로 셋째는 싱크의 이동성을 지원함에 있어 통신의 강건성 측면에서 ARBITER를 TTDD, DSM[7]과 비교하는 것이다.

4.1 시뮬레이션 환경과 평가 항목

우리는 QualNet simulator[15]를 사용하여 제안 방안을 시뮬레이션 했다. 표 1은 시뮬레이션의 구체적인 설정을 나타낸다. 센서 노드 하나의 전송, 수신, 대기 전력 소비율은 각각 21mW, 15mW, 0.03mW 이다. 장치 설정은 MICA 명세서[16]를 참조하여 선택했다. 또한 저장 노드와 전달 노드의 배치는 그림 2와 같으며 α는 100m로 고정했다.

평가 항목을 설명하기 전에, 하나의 시뮬레이션 용어를 정의한다. 완료 시간 (round time)은 망 초기화 과정을 포함하여 모든 싱크가 쿼리를 신청하고 모든 소스로부터 쿼리에 대한 정보를 받는 시간을 말한다. 우리는 ARBITER의 성능을 분석하고 비교하기 위한 다섯 가지의 평가 항목을 선택했다.

*총 에너지 소모량*은 한 완료 시간 동안 모든 센서 노드들의 총 에너지 소모량의 평균으로 정의된다. 대기 상태의 에너지는 데이터 생성 간격과 관련이 있으며 데이터 전송 효율성을 나타내지 않기 때문에 포함하지 않는다. 이 평가 항목의 값이 0이면 모든 센서 노드들이 더 이상 에너지를 소모하지 않았다는 것을 뜻하며, 그것은 시스템의 센서 네트워크의 수명이 끝났다는 것을 말한다.

*평균 전송률*은 센서 노드들에 의해 생성된 보고들의 총 개수에 대한 모든 싱크들이 성공적으로 수신한 보고들의 총 개수 비율로 측정한다. 싱크들과 소스들의 개수에 따른 결과를 비교하기 때문에, 한 개에서부터 스무 개까지의 싱크들과 소스들을 시뮬레이션 지역에 랜덤하게 위치시켰다.

*평균 응답 시간*은 저장 노드가 어떤 소스로부터 데이터를 수신한 상태라고 가정했을 때, 싱크가 쿼리를 저장노드에게 보낸 시점에서부터 싱크가 저장 노드로부터 그 쿼리에 대한 정보를 수신하는 시점

표 1. 시뮬레이션 환경 설정

Parameters	Values
Simulation network space	1000m × 1000m
The number of nodes	200 nodes
Node placement	Uniform deployment
Transmission range	Uniform 55m
Sink mobility model	Random way point
Sink speed	10 m/s
The number of sinks and sources	1 through 20

까지의 평균 대기 시간으로 측정한다. 시뮬레이션에서는 소스가 한 완료 시간 안에서 랜덤한 시간에 데이터를 생성한다.

*정보 전송률*은 최초 소스가 전송한 정보 패킷들의 개수에 대한 모든 싱크들에게 성공적으로 도착한 정보들의 개수 비율로 정의한다.

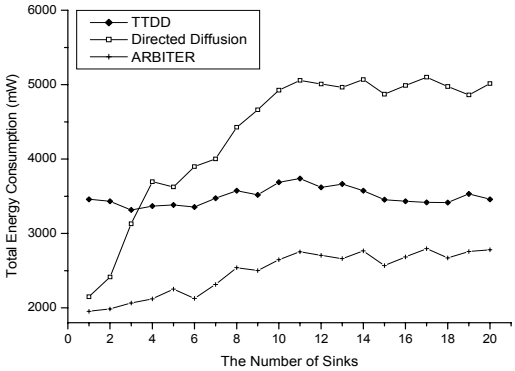
*정보 전송 지연 시간*은 하나의 저장 노드가 하나의 정보 패킷을 모든 싱크들에게 전송하는 순간으로부터 모든 싱크들이 그 정보 패킷을 수신하는 모든 순간들 사이의 평균 대기 시간으로 측정한다. TTDD의 경우, 우리는 하나의 소스와 모든 싱크들로 고려했다. DSM의 경우는 하나의 싱크와 모든 사용자들로 고려했다.

4.2 시뮬레이션 결과

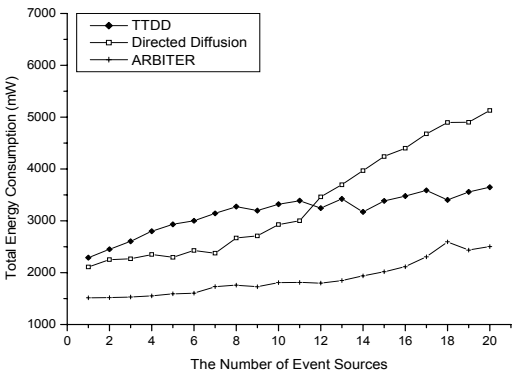
4.2.1 확장성에 대한 비교

그림 4의 (a), (b), (C)는 확장성의 기능 요소로써 싱크들과 소스들의 수에 따른 총 에너지 소모량과 싱크들의 수에 따른 평균 전송률을 나타내는 그래프들이다. 우리는 이 시뮬레이션을 싱크 수 변경과 소스 수 변경의 각각 두 가지 상황으로 실행하였다. 첫 번째 상황으로 소스의 수를 20으로 고정하고 싱크 수의 변화에 따른 총 에너지 소모량과 평균 전송률의 변화를 분석했다. 두 번째 상황으로 싱크 수를 20으로 고정하고 소스 수의 변화에 따른 총 에너지 소모량의 변화를 분석했다.

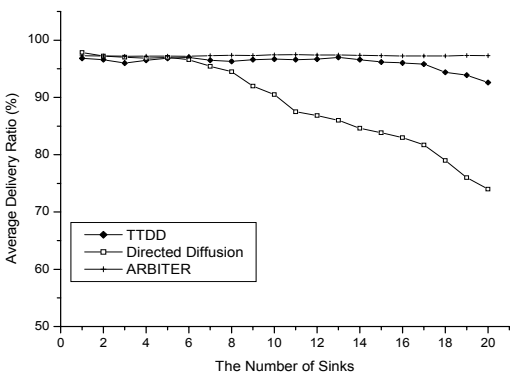
그림 4(a)에서 볼 수 있듯이 ARBITER는 싱크 수의 변화에 따라 정보의 통신 즉, 쿼리 전송, 데이터 저장, 데이터 전송을 위한 총 에너지 소비량이 매우 천천히 증가한다. TTDD가 거의 안정된 성능을 보인다 하더라도, TTDD는 ARBITER보다 대략 1000mW에서 1500mW 더 많은 에너지 소비량이 필요하다. 이것은 TTDD가 격자 구조체를 각 소스마다 구성하기 때문이다. 이 시뮬레이션에서는 20개의 구조체가 생성됐다. Directed Diffusion에서는 모든 싱크들이 자신의 쿼리를 모든 센서 필드에 플러딩하며, 모든 소스들은 다중 경로를 통해 싱크에게 데이터를 전송한다. 우리의 시뮬레이션 환경상에서 모든 싱크들은 필드 상에 랜덤하게 분산되어 있으며 20개의 소스들이 분산되어 있는 모든 싱크들에게 다중 경로를 통해 데이터를 전송하기 때문에, 싱크 수에 따라 총 에너지 소비량이 급격하게 증가하는 좋지 않은 성능을 보였다. 싱크 수를 고정하고 소스 수를 증가시키는 상황에서도 ARBITER는 TTDD



(a) 싱크 수에 따른 총 에너지 소모량



(b) 소스 수에 따른 총 에너지 소모량



(c) 싱크 수에 따른 평균 전송률

그림 4. 확장성에 대한 비교

와 Directed Diffusion에 비해 총 에너지 소모량 면에서 좋은 성능을 보였다. 이는 ARBITER가 데이터 융합과 멀티캐스팅을 지원하며 쿼리 전달과 데이터 수집을 위한 격자 구조체를 오직 하나만 만들기 때문이다. 그림 4(b)에서 알 수 있듯이 TTDD와 Directed Diffusion의 경우, TTDD가 소스 수 변화에 따른 총 에너지 소모량에서 더 나은 성능을 보

인다. 이는 소스들이 다중 경로로 데이터를 전송하는 Directed Diffusion과 달리, TTDD는 격자를 생성하려는 소스들이 매우 가까이 위치한다면 격자 구조체를 공유할 수 있기 때문이다. Directed Diffusion의 이런 특징은 그림 4(c)에서 볼 수 있듯이 싱크 수에 따른 평균 전송률에도 영향을 미친다. 20개의 소스들로부터의 다중 경로를 통한 데이터 전송은 충돌 확률을 매우 높이기 때문에 평균 전송률을 낮춘다. 이 시뮬레이션 결과는 ARBITER가 TTDD와 Directed Diffusion에 비해 더 나은 확장성을 제공할 수 있다는 것을 의미한다.

4.2.2 데이터 신뢰도에 대한 비교

그림 5는 싱크가 관심 데이터를 요청한 후 센서 네트워크로부터 정보를 수신할 때 평균 응답 시간을 보여준다. 이 실험은 다음과 같은 환경에서 이루어졌다. 20개의 소스와 1개의 싱크가 랜덤하게 배치됐고, 쿼리 메시지와 데이터는 한 완료 시간 동안 매초마다 생성했다. TTDD에서 소스는 데이터가 발생하지 않으면 격자 구조체를 생성하지 않기 때문에, 싱크는 두 번째 쿼리부터 데이터를 수신할 수 있다. 따라서 그림 5에서 볼 수 있듯이 TTDD는 싱크가 쿼리를 하는 시간으로부터 20개의 격자 구조체가 만들어지는 시간까지 포함하여 대략 3초 정도의 시간 후에 데이터를 수신할 수 있다. Directed Diffusion과 ARBITER는 싱크가 생성하는 쿼리를 통해 정보를 요청한다. 그러나 Directed Diffusion은 플러딩을 통해 센서 필드 전체에 쿼리를 하기 때문에 응답 시간이 ARBITER보다 느려진다. ARBITER에서 싱크는 전달 노드들의 중계와 위치 기반 포워딩을 통해 중앙에 있는 저장 노드에게 직접적으로 정보를 요청하며, 저장 노드는 소스에서 발생한 데이터를 버퍼링을 통해 저장 관리한다. 따라서 ARBITER는 Directed Diffusion에 비해 70%정보 빠른 응답 시간을 제공한다. 즉, 이 성능 향상은 공표/신청 패러다임의 시간 분리성 지원에서 비롯된 것이다.

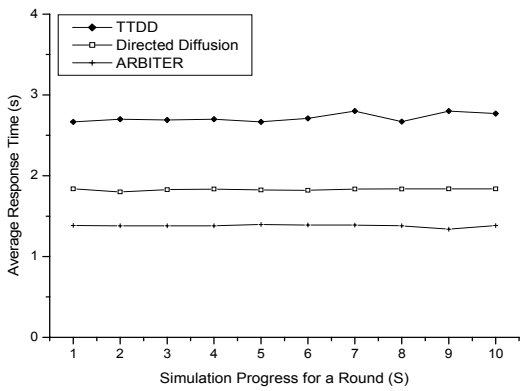
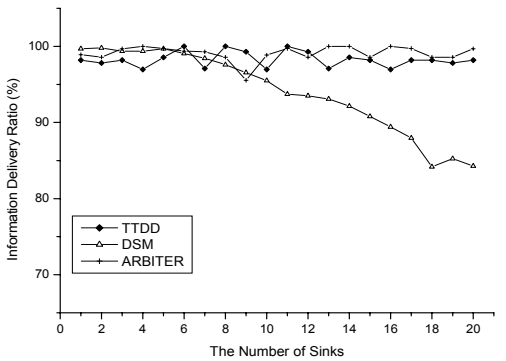


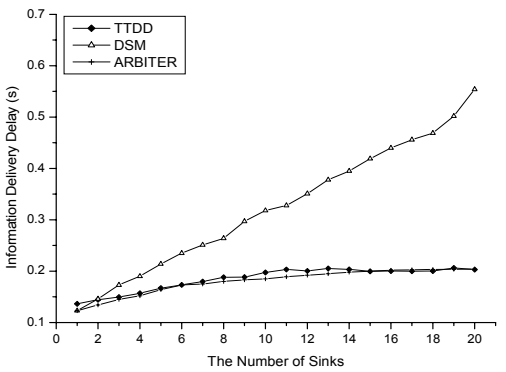
그림 5. 평균 응답 시간

4.2.3 이동성 지원에 대한 비교

이 실험에서 우리는 소스의 개수를 20개로 고정하고, 사람의 걷는 속도를 반영한 10m/s의 속도로 움직이는 싱크의 개수에 따른 TTDD, DSM, ARBITER의 이동성 지원 능력에 대해 분석했다. ARBITER의 경우 저장 노드가 싱크들에게 정보 패킷을 전송하는 순간부터 모든 싱크가 정보 패킷을 수신하는 모든 순간까지의 평균 대기 시간을 측정했다. TTDD의 경우 소스 노드가 모든 싱크들에게 정보 패킷을 전송하는 순간부터 모든 싱크들이 정보 패킷을 수신하는 모든 순간들까지의 평균 대기 시간을 측정했다. DSM은 경우는 싱크가 모든 이동 사용자들에게 정보 패킷을 전송하는 순간부터 모든 사용자들이 정보 패킷을 수신하는 모든 순간들까지의 평균 대기 시간을 측정했다. 그림 6(b)는 앞서



(a) 정보 전송률



(b) 정보 전송 지연 시간

그림 6. 이동성 지원에 대한 비교

언급한 방법들로 측정된 정보 전송 지연 시간을 나타낸다. ARBITER와 TTDD는 정보를 멀티캐스팅으로 제공하기 때문에, 싱크들의 수가 많아진다고 해도 0.12초에서 0.2초의 낮은 지연 시간을 유지한다. DSM은 유니캐스트 방법으로만 정보를 전송하기 때문에 싱크의 수가 많아질수록 평균 지연 시간이 증가한다.

그림 6(a)는 이동 싱크나 유저가 이동 중에 정보를 잘 수집할 수 있는가에 대한 지표로써 정보 전송률을 나타낸다. ARBITER와 TTDD는 거의 100%를 유지한다. 그러나 DSM의 경우 모든 싱크마다 충돌율이 높아지는 원인이 되는 쿼리 플러딩을 통해 구조체를 생성하며, 유니캐스트로 정보 전달이 이루어지기 때문에 정보 전송률이 떨어진다.

Ⅴ. 결 론

우리는 싱크와 소스에게 분리성을 제공하는 새로운 통신 방안인 ARBITER를 제안했다. ARBITER는 세가지 분리성 특성을 제공할 수 있었고, 따라서 대규모 무선 센서 네트워크에 확장성과 더 나은 이동성 지원을 제공할 수 있었다. ARBITER는 또한 가변적 라우팅과 멀티캐스팅을 지원함으로써, 에너지 소모량과 통신 비용을 절감해 네트워크 수명을 증가시킬 수 있었다. 우리는 ARBITER를 확장성, 데이터 신뢰도, 이동성 지원 측면에서 다른 방안들과 비교했다. 우리의 시뮬레이션 결과는 ARBITER가 더 나은 성능을 보인다는 것을 증명했다.

참 고 문 헌

〔1〕 I.F. Akyildiz et al., “A survey on sensor networks,” *IEEE Communications Magazine*, Vol. 40, pp. 102-114, Aug., 2002.

〔2〕 H. Karl and A. Willing, *Protocol and Architecture for Wireless Sensor Networks*, John Wiley & Sons, Inc., 2005

〔3〕 B. Krishnamachari et al., “Modeling Data-Centric Routing in Wireless Sensor Networks,” *IEEE INFOCOM* 2002.

〔4〕 P. Th. Eugster et al., “The Many Faces of Publish/Subscribe,” *ACM Computing Surveys (CSUR)*, 2003

〔5〕 Y. Huang and H. Garcia-Molina, “Publish/subscribe in a mobile environment,” *Wireless Networks*, Vol.10, Iss., Nov., 2004, pp.643-652.

〔6〕 C. Intanagonwiwat et al., “Directed Diffusion for Wireless Sensor Networking,” *IEEE/ACM Transactions on Networking*, Feb., 2003.

〔7〕 S. Park et al, “A Communication Architecture to Reflect User Mobility Issue in Wireless Sensor Fields,” *IEEE WCNC*, Mar., 2007.

〔8〕 F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang, “A Two- Tier Data Dissemination Model for Large-scale Wireless Sensor Networks,” *ACM MobiCom*, Sep., 2002.

〔9〕 J. H. Shin et al., “RailRoad: Virtual Infrastructure for Data Dissemination in Wireless Sensor Networks,” *ACM International Workshop PE-WASUN*, 2005.

〔10〕 E. B. Hamida and G. Chelius, “A Line-Based Data Dissemination Protocol for Wireless Sensor Networks with Mobile Sink,” *IEEE ICC*, China, May, 2008.

〔11〕 T. He et al., “Range-free localization schemes for large scale sensor networks,” *ACM MobiCom* 2003, pp.81-95.

〔12〕 B. Hofmann-Wellenhof et al., *Global Positioning System: Theory and Practice*, Springer, 4th edition, 1997.

〔13〕 S. Ratnasamy et al., “GHT: A Geographic Hash Table for Data-Centric Storage in Sensornets,” *ACM International Workshop WSNA*, Sep., 2002.

〔14〕 B. Karp and H. T. Kung. “GPSR: Greedy perimeter stateless routing for wireless networks,” *ACM MobiCom*, Aug., 2000.

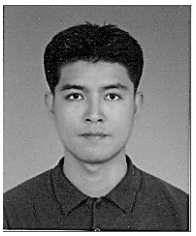
〔15〕 Scalable Network Technologies, Qualnet, [online] available: http://www.scalable- networks.com

〔16〕 J.Hill and D. Culler, “Mica: a wireless platform for deeply embedded networks,” *IEEE Micro*, Nov./Dec., 2002.

박 수 창 (Soochagn Park) 정회원

2005년 8월 충남대학교 정보통신공학부 컴퓨터 전공
2007년 8월 충남대학교 컴퓨터공학과 석사
2007년 9월~현재 충남대학교컴퓨터 공학과 박사과정
<관심분야> Internet Routing, Wireless Sensor Networks, MANET, 4G, Multicast 등

이 의 신 (Euisin Lee) 정회원



2005년 8월 충남대학교 정보통신공학부 컴퓨터 전공
2007년 8월 충남대학교 컴퓨터공학과 석사
2007년 9월~현재 충남대학교컴퓨터 공학과 박사과정
<관심분야> Internet Routing, Wireless Sensor Networks, MANET, Geographic Routing, Mobility 등

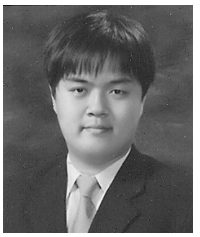
박 호 성 (Hosung Park) 준회원

2008년 2월 충남대학교 정보통신공학부 컴퓨터 전공
2008년 3월~현재 충남대학교컴퓨터 공학과 석사과정
<관심분야> Internet Routing, Wireless Sensor Networks, MANET, Mobility, Reliability 등

이 정 철 (Jeongcheol Lee) 준회원

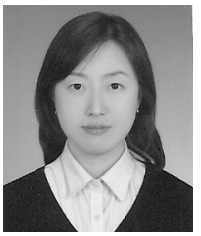
2008년 2월 충남대학교 정보통신공학부 컴퓨터 전공
2008년 9월~현재 충남대학교컴퓨터 공학과 석사과정
<관심분야> Internet Routing, Wireless Sensor Networks, Mobility, Multicasting 등

오 승 민 (Seungmin Oh) 준회원



2009년 2월 충남대학교 정보통신공학부 컴퓨터 전공
2009년 3월~현재 충남대학교컴퓨터 공학과 석사과정
<관심분야> Internet Routing, Wireless Sensor Networks, MANET, Mobility 등

정 주 현 (Juhyun Jung) 준회원



2000년 2월 건양대학교 컴퓨터공학과
2009년 3월~현재 충남대학교컴퓨터 공학과 석사과정
<관심분야> Internet Routing, Wireless Sensor Networks, MANET, Mobility, Real-time 등

김 상 하 (Sang-Ha Kim) 종신회원

1980년 서울대학교 학사
1984년 University of Huston 석사
1989년 University of Huston 박사
1992년~현재 충남대학교 정보통신공학부 교수
<관심분야> Internet Routing, Wireless Sensor Networks, MANET, 4G, Mobility, Multicast 등