Journal of Korea Multimedia Society Vol. 19, No. 10, October 2016(pp. 1808-1815)
http://dx.doi.org/10.9717/kmms.2016.19.10.1808

에너지 효율적인 무선 네트워크용 상호 시각 동기화 프로토콜

배 시 규†

An Energy-efficient Pair-wise Time Synchronization Protocol for Wireless Networks

Shi-Kyu Bae†

ABSTRACT

TPSN(Timing-sync Protocol for Sensor Networks), the representative of time synchronization protocol, has been already developed to provide time synchronization among nodes in wireless sensor networks. Even though the TPSN's method has been referenced by so many other time synchronization schemes for resource-constrained networks like wireless sensor networks or low power personal area networks, it has some inefficiency in terms of power consumption and network-wide synchronization time (or called convergence time). The main reason is that each node in TPSN needs waiting delay to solve the collision problem due to simultaneous transmission among competing nodes, which causes more power consumption and longer network convergence time for a network-wide synchronization. In this paper an improved scheme is proposed by changing message exchange method among nodes. The proposed scheme not only shortens network-wide synchronization time, but also reduce collision traffic which lead to needless power consumption. The proposed scheme's performance has been evaluated and compared with an original scheme by simulation. The results are shown to be better than the original algorithm used in TPSN.

Key words: Time Synchronization, TPSN(Timing-sync Protocol for Sensor Networks), Wireless Sensor Network, Contention, Collision, Polling.

1. 서 론

무선 센서 네트워크(WSN : Wireless Sensor Networks)의 활용성과 잠재력이 예상보다 크다는 것을 알게 되고 여러 관련분야에서 활발한 연구가 이루어지고 있으며, 특히 사물인터넷에서도 중요한 역할을 하게 되었다. 무선네트워크를 구성하는 각 노드는 응용에 따라 다른 노드와 공통 클럭을 유지하는 것이 중요하므로 네트워크 전체 혹은 일부 노드간에 시각 동기화가 필요하게 된다. 특히, 멀티미디어 전송을 위해서는 미디어내 혹은 미디어간의 동기화가 중요하며, 이는 여러 노드간의 정확한 시각 동기화를 필요로 한다.

시각 동기화 기술은 인터넷 초기 시절부터 NTP (Network Time Protocol)[1]를 대표로 하여 전통적으로 행해져 왔으나, 무선 네트워크 환경, 특히 무선 센서 네트워크 분야에서의 새로운 요구사항을 고려하여 다시 연구되고 있다. 무선 센서 네트워크와 같이 자원이 충분하지 않은 노드들로 구성된 네트워크를 설계할 때 공통적으로 전력 소모를 최소화하는

※ Corresponding Author : Shi-Kyu Bae, Address: (750-711) Dongyangdae-ro 145, Pung-Gi, Youngju, Gyoung Buk, Korea, TEL : +82-54-630-1071 FAX : +82-54-636-8523, E-mail : skbae@dyu.ac.kr

Receipt date : Aug. 3, 2016, Revision date : Sep. 21, 2016
Approval date : Oct. 24, 2016
† Dept. of Computer Eng., DongYang University

것이 중요하며, 시각 동기화 프로토콜을 설계할 때에도 마찬가지로 전력 소모를 최소한으로 줄이는 것이 필요하다[2].

무선센서네트워크에서 시각 동기화 문제를 해결하기 위하여 여러 프로토콜이 개발되었는데, TPSN (Timing-sync Protocol for Sensor Networks)[3]도 그 중에 하나이며, NTP를 기반으로 한다. TPSN은 동기화 전에 레벨 탐색이라는 과정을 통해 계층 구조를 생성하며, 이 계층 구조를 사용하여 네트워크의 전체 노드를 순차적으로 동기화한다. 전체 네트워크를 동기화하는 과정에서 트리의 낮은 레벨에 위치하는 센서 노드들은 한 레벨 높은 위치의 노드와 패킷을 교환하며, 이런 방식을 [2]에서는 상호(Pair-wise) 동기화 방식으로 분류하였다. 한 노드를 기준으로 (부모 노드) 하위 레벨의 노드(자식 노드)가 여러 개 있을 때 부모 노드와 자식 노드들 간의 패킷 교환은 서로 경쟁하게 되고 충돌이 발생할 수 있다. 여러 노드들이 패킷을 전송하는 과정에서 충돌이 발생하면 전체 네트워크를 동기화하는데 걸리는 시간이 길어질 뿐 만 아니라 더 많은 전력이 소모된다.

이 논문에서는 무선 네트워크에서 부모 노드와 자식 노드들간의 데이터 교환 방식을 변경하여 전송 데이터 량과 충돌을 줄이고 동기화 완료 시간을 감소시키는 에너지 효율적인 시각 동기화 방식을 제안한다. 논문의 구성은 다음과 같다. 2장에서 기존의 무선 네트워크용 시각 동기화 프로토콜과 TPSN의 동작을 살펴본다. TPSN의 문제점을 정리하고 개선된 TPSN 알고리즘을 3장에서 제안한다. 4장에서 제안한 알고리즘을 적용하여 성능을 분석하며, 마지막으로 5장에서 결론을 다룬다.

2. 관련 연구

2.1 WSN 시각 동기화 프로토콜

지금까지 개발된 여러 시각 동기화 프로토콜들은 송신자와 수신자간의 메시지를 주고받는 형태에 따라 송신-수신형과 수신-수신형으로 나뉜다. 송신-수신형 동기화는 한 노드가 송신자가 되어 데이터를 전송하며, 송신자를 제외한 다른 노드들은 수신자가 되어 보내어진 데이터를 수신하며, TPSN[3]과 FTSP (Flooding Time Synchronization Protocol)[4]이 여기에 속하고, 글로벌 참조 클럭을 전체 네트워크에 전달할 수 있다. 이와 대조적으로 수신-수신형 동기화에서는 참조 노드가 특정한 패킷을 한 번 전송하면, 이 후에는 전송한 노드를 제외한 다른 노드들이 서로 협력하여 동기화를 이루는 방식을 갖는다. 이 종류에 해당하는 대표적인 프로토콜이 RBS(Reference Broadcast Synchronization) [5]로서, 한 노드(송신자)가 다른 수신 노드들이 동시에 수신하도록 비이컨(beacon) 메시지를 방송하면, 송신자를 제외한 다른 수신자들이 이 비이컨 메시지를 수신한 각각의 로컬 클럭 값을 서로 교환하여 상호간의 클럭 오프셋을 기록한다. 이 RBS 프로토콜은 가까운 거리에 위치한 여러 노드간에 상대적인 클럭 오프셋을 비교적 정확하게 파악할 수 있지만, 특정 노드의 클럭(혹은 글로벌 클럭)으로 네트워크 전체를 동기화할 수는 없다. RBS와 FTSP의 장점을 결합한 혼합형 방안[6]이나, TPSN의 성능을 개선하기 위한 여러 제안들이 있었다[7-10]. [7]의 논문에서는 클러스트의 레벨을 제한시켜 트리의 깊이를 줄임으로써 성능을 개선시키는 방안을 발표하였다. [8]의 저자들은 자식 노드가 가장 낮은 레벨의 노드를 부모 노드로 선택하는 방안을 제안하였다. [9]의 저자는 TPSN에서 부모 노드와 자식노드간의 메시지 교환을 경쟁방식이 아닌 폴링 방식으로 구현하여 전력 소모를 개선하였다. [10]에서는 2회 메시지 교환을 여러 번 실시하여 클럭 오프셋뿐 아니라 클럭 스큐를 측정하여 동기화에 반영하였다.

2.2 TPSN[3]의 동작

TPSN은 전형적으로 트리 구조에서 동작하며, 루트 노드로부터 시작하여 상위 레벨의 노드와 연결된

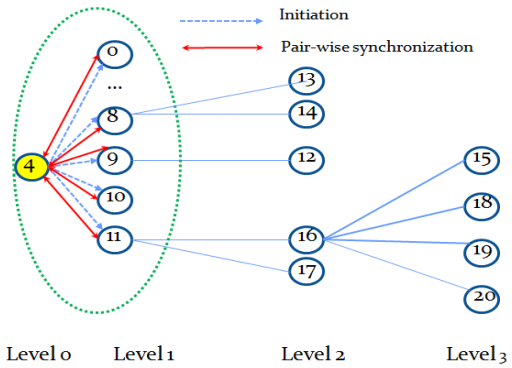

Fig. 1. Operation of TPSN [2].

하위 레벨의 노드들 간의 메시지 교환을 통해 동기화를 이루고, 순차적으로 전체 네트워크를 동기화한다. Fig. 1에 나타난 것처럼 루트 노드(Level 0로 표시)와 연결된 Level 1 노드 간에 동기화가 이루어진 후, 연속적으로 Level 1 노드와 연결된 Level 2 노드들 간에 동기화가 이루어져, 최종적으로 네트워크의 모든 노드가 동기화가 된다.

TPSN의 동작을 좀 더 구체적으로 살펴보면, TPSN은 두 개의 단계, 즉, 레벨 탐색 단계와 동기화 단계로 이루어진다. 레벨 탐색 단계에서는 트리 구조가 생성되며, 트리의 생성은 플러딩(flooding)을 약간 변형한 방식으로 이루어진다. 동기화 단계에서 모든 노드는 상위 레벨의 노드와 양방향 통신을 통해서 동기화를 이루며 최종적으로 모든 노드는 루트 노드와 동기화를 이룬다. 각 레벨에서 부모 노드와 연결된 자식 노드 간에는 메시지 교환이 이루어지는데, 부모노드의 시작 메시지를 시작으로 각 자식 노드가 적절한 시기에 부모 노드에게 동기화 요청 메시지를 보내고 이에 대한 응답을 부모노드로부터 받음으로서 그 노드의 동기화가 이루어진다. 동기화가 완료된 노드는 다시 연결된 자식 노드와 동기화를 시도한다. 이 때, 여러 노드가 한 부모 노드와의 메시지 전송을 동시에 시도하면 충돌이 일어나므로, TPSN에서는 각 노드에서 전송을 시도하기 전에 임의의 대기 시간을 갖도록 하고 있다.

3. 제안 방안

TPSN에서 적용하고 있는 각 노드의 임의의 대기 시간은 적정 값을 정하기 어려우므로 충돌을 완전히 해결할 수 없어 충돌에 의한 대기 지연 시간 및 재전송 등으로 인한 추가적인 네트워크 동기화 지연과 전력 소모의 오버헤드가 발생한다. 설사 충돌이 일어나지 않더라도 이 대기 시간은 각 노드의 동기화 완료 시간을 지연시키고 최종적으로 전체 네트워크의 동기화 완료 시간에 영향을 끼치며, 네트워크 동기화 완료 시간이 길어짐으로서 이에 따라 전력 소모도 커지게 된다. 네트워크 전체 동기화가 완료되기 전에 발생하는 이벤트의 시각 정확도는 응용에 영향을 끼치게 되므로, 네트워크 전체 동기화 완료 시간은 가능하면 작을수록 좋다.

이 논문에서 제안하는 방안은 TPSN의 동작에서 부모 노드와 자식 노드간의 메시지 교환을 경쟁이 아닌 폴링 방식을 사용한다. 시각 동기화에서 한 부모 노드와 연결된 자식 노드간의 데이터 전송은 센서 데이터 전송이나 라우팅 등의 일반적인 데이터 전송과는 다른 특징을 가지고 있다. 즉, 일반 노드간의 데이터 전송은 언제 어느 노드 간에 전송이 발생할지 예상할 수 없지만, 네트워크 전체 노드의 시각을 동기화하는 동작은 한 부모 노드를 중심으로 연결된 모든 자식 노드들이 반드시 일정한 규칙에 의해 데이터 전송에 교대로 참여한다. 그러므로 경쟁 방식보다는 시분할이나 폴링방식이 더 적합하다. 시분할 방식은 트리가 구성될 때 한 부모 노드가 자식 노드의 수를 파악하여 각 자식 노드에게 일정한 시간 슬롯을 할당하여 정해진 시간 내에서만 전송을 하도록 하는 것으로, 시간 슬롯의 정확도는 노드간의 동기화 정확도에 영향을 받는다. 이에 비하여 폴링 방식은 간단하고 노드간의 시간 정확도에 영향을 받지 않으므로 3가지 방식 중 가장 적합하다. 이러한 이유로 [9]의 제안에서는 부모노드와 여러 자식 노드간의 데이터 교환을 경쟁 방식에서 폴링 방식으로 변경하였다. 이렇게 함으로서 각 노드의 대기 시간을 줄이고 충돌을 예방하여 전력 소모와 동기화 완료 시간을 개선하였지만, 노드간에 주고 받은 메시지의 횟수(또는 전송 데이터 크기)는 변함이 없었다.

이 논문에서는 상하위 레벨의 노드 간에 데이터 전송을 [9]와는 다른 분산 폴링 방식으로 하면서 추가로 전송 메시지의 회수(또는 전송 데이터 크기)를 줄이는 방안을 제안한다. 이렇게 함으로서 전체 전송 데이터의 량을 더욱 감소시키고 각 노드가 전송 전에 대기하는 시간을 줄여 전력 소모와 동기화 완료 시간을 개선할 수 있다. 이 논문에서 제안된 방식을 EPSP (Energy-efficient Pair-wise Synchronization Protocol)이라고 부르기로 하며, TPSN과 유사하게 트리 구성과 동기화의 두 개의 동작 상태를 갖는다.

3.1 EPSP의 동작: 트리 구성

TPSN의 탐색 과정에서의 트리 구성은 간단한 플러딩(flooding) 방식을 사용하고 임의의 대기 시간으로 인하여 무작위로 생성되므로, 생성된 트리에서 동기화를 실행하면 레벨 수가 증가할수록(즉, 트리의 깊이가 클수록) 루트 노드를 기준으로 한 말단 노드의 클럭 오차는 점점 커지며, 전체 노드의 평균 동기화 오류도 증가하게 된다. 따라서 트리의 구성은 깊

```
Set Radius to R1 : radius of broadcast range
if (root node)
Broadcast a discovery-level packet (node-id, level#)
For all nodes except root node {
  if (packet received && firstly received) {
    Set level=parent node's level+1;
    Distance = measure distance
        between parent node and node itself;
    Wait (k*Radius/distance) seconds;
    Broadcast additional discovery-level packet
        including its id to notify the parent;
 // Send its id to the parent to notify a real child;
  }
}
```

Fig. 2 Tree Construction Algorithm.

이가 얕을수록 동기화 오류가 개선된다. Fig. 1에서 예를 들면, 레벨 2에 있는 노드 12와 노드 16은 노드 레벨 1에 있는 노드 0이나 노드 11보다 덜 정확하고, 가장 낮은 레벨에 있는 노드 15나 노드 20이 가장 오류가 크다.

EPSP에서는 TPSN에서처럼 불규칙적이지 않고 동기화에 더 효율적인 트리를 구성하기 위하여 [11]에서 제안한 트리 구성 알고리즘과 유사한 트리 구성 알고리즘[9]을 사용하되, EPSP에 적합하게 수정하였다. 트리 구성 알고리즘이 실행되면 Fig. 1과 같은 트리가 구성되는데, 루트 노드를 포함한 각 부모 노드는 자식 노드의 정보를 가지게 되며, 말단에 있는 노드(leaf nodes)는 자식 노드로부터 통지를 받지 않으므로 부모 노드가 되지 않는다. Fig. 1에서 예를 들면, 노드 12, 노드 17, 노드 15 등은 부모 노드가 아니다.

원래 TPSN의 동작에서는 최초로 메시지를 전달한 노드를 전달받은 노드에서 부모노드로 인식하고 부모 노드의 ID를 기억한다. 하지만, EPSP에서는 자식노드 뿐 아니라 부모 노드도 자신의 자식 노드들의 ID 리스트를 알고 있어야 하므로, 트리 형성 과정에 이 부분이 필요하다. 그리고 TPSN의 트리 구성과 가장 차이가 나는 부분은 트리 구성 시 적용하는 각 노드의 전송 범위와 동기화 동작에서 데이터 전송할 때 적용하는 전송 범위가 다르다는 것이다. 즉, EPSP가 동작하기 위한 조건은 TPSN이나 [9]의 제안과는 다르며, 기존의 방식으로 구성한 트리를 사용할 수 없으므로, 새로운 트리 구성 방법이 필요하다. 자세한 내용은 다음의 동기화 동작에서 설명한다.

3.2 EPSP의 동작 : 동기화 단계

부모 노드와 자식 노드사이의 동기화는 다음과 같이 동작한다. TPSN에서 부모 노드가 동기화 시작을 자식 노드들에게 통지하면, 각 자식 노드들은 임의의 지연 시간 후 부모 노드와 메시지 교환을 시작한다. 각 자식 노드는 부모 노드에게 요청 메시지를 보내고 부모 노드로부터 응답 메시지를 받으면 메시지에 포함된 시각 정보를 사용하여 부모노드에 대하여 동기화한다. 동기화가 완료된 노드들은 다시 자식 노드들에게 부모 노드 역할을 하여 동기화를 연속적으로 실시한다. 모든 말단 노드까지 동기화가 끝나면 네트워크 전체의 동기화가 완료된다. 이 과정에서 각 노드들은 인근 노드들과의 충돌을 겪게 되고 재전송함으로써 추가로 전력 소비와 지연 시간이 증가한다.

EPSP에서는 TPSN과 달리 각 부모 노드가 자식 노드들에게 동기화 시작을 알리면 각 자식 노드가 순차적으로 부모 노드에게 동기화 요청 메시지를 보내고 모든 요청 메시지를 받은 후 이에 대한 응답을 부모 노드가 한꺼번에 모든 자식 노드들에게 전송한다. 이렇게 하면 한 부모 노드와 연결된 자식 노드간의 동기화를 위한 전송 메시지 수는 줄어든다.

Fig. 3-(a)는 부모노드와 자식 노드간의 동기화 메시지 교환 시의 시간 관계를 나타낸 것이다. 부모 노드가 “Initiate” 패킷을 보내어 자식 노드들이 순서대로 “REQ” 패킷을 보내도록 한다. 이때 "Initiate" 패킷에는 부모 노드에 연결된 모든 자식 노드의 리스트를 함께 보내어 각 자식 노드들이 자신 외의 다른 형제 노드들의 정보를 알 수 있도록 한다.

단계별 동작을 구체적으로 설명한다. 가령, 현재의 부모노드에 연결된 자식 노드 1,2,3이 있다고 하자.

1) 부모 노드가 "Initiate" 패킷으로 자식 노드 1,2,3의 정보를 모든 자식 노드에게 전송한다.

2) 자식 노드 리스트의 가장 앞에 있는 1번 노드가 가장 먼저 동기화 요청 메시지를 전송한다.

3) 다음 2번 노드는 1번 노드가 요청 메시지를 보냈다는 것을 overhearing하여 알게 되며, 다음이 자신의 차례이므로 즉시 요청 메시지를 부모 노드에게 전송한다.

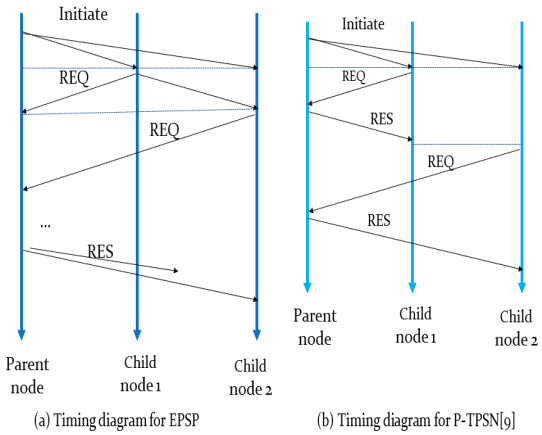

Fig. 3. Timing diagrams for message exchanges among parent and child nodes in (a) EPSP, and (b) P-TPSN.

4) 나머지 자식 노드도 순서대로 요청 메시지를 전송한다.

5) 부모 노드는 모든 자식 노드로부터 요청 메시지를 받았음을 확인하고, 각 자식 노드들에게 보낼 응답 메시지를 묶어서 한꺼번에 응답 메시지로 보낸다.

6) 각 자식 노드는 부모 노드로부터 받은 응답 메시지 정보를 해석하여 자신의 시각 정보를 부모 노드와 동기화한다.

7) 이로써 한 부모 노드와 연결된 자식 노드들 간의 동기화가 완료된다. 각 자식 노드들 중 부모 노드로 역할 하는 노드는 반복해서 자신의 자식 노드들과 앞의 동작을 계속한다.

8) 네트워크의 모든 노드가 동기화를 완료하면 하나의 동기화 라운드가 종료된다.

[9]에서 제안된 P-TPSN은 중앙 집중형 폴링 방식으로 부모 노드와 자식 노드들 간에 메시지 교환이 이루어지는 면에서는 EPSP과 공통점이 있으나 다음과 같은 차이점이 있다.

1) P-TPSN에서 각 자식노드는 TPSN과 마찬가지로 2-way handshake로 동작하며, 부모 노드의 지시에 따라 자신의 순서에 요청 메시지를 시도한다. 하지만, EPSP에서는 자식 노드들의 전송 메시지를 overhearing하여 자신의 전송 순서와 시점을 결정하여 요청 메시지를 전송한다. 즉, 다른 자식노드의 요청 메시지에 대한 부모 모드로부터의 응답 메시지를 수신할 필요가 없으므로, 메시지 전송 생략에 따른 소비 전력과 지연 시간이 감소한다. Fig. 3-(b)는 P-TPSN에서의 부모노드와 자식 노드간의 동기화 메시지 교환 시의 시간 관계를 나타낸 것이다.

2) EPSP에서 자식 노드들 간에 overhearing이 가능하려면 서로 전송 범위 내에 위치하여야 한다. 즉, EPSP가 동작하기 위한 조건은 TPSN이나 [9]의 제안과는 달라 기존의 트리 구성 알고리즘은 사용할 수 없고, 새로운 트리 구성 방법이 필요하다.

트리를 T라고 하고, 각 노드의 집합을 V, 두 노드 간의 연결 링크를 E라고 할 때, T = (V, E)로 나타내기로 하자. 노드 i 의 자식 노드 집합 C는 노드 i로부터 통신 반경 내에 있어 연결되고 다음과 같이 정의할 수 있다.

$$C(i)=\{i,j,\ k\in V : (i,j)\in E,\ (i,k)\in E\ \} \quad (1)$$

노드 j의 부모 노드를 P(j)라고 하면, 식 (1)에서 P(j) = i , P(k) = i이다.

EPSP에서는 추가로 모든 자식 노드가 전송 범위 내에 존재하여 서로 연결되어야 하는 조건이 필요하다. 즉, (j, k) ∈ E 가 만족되어야 한다.

EPSP의 동기화 단계에서 사용할 전송 범위를 R이라고 하고, 트리를 구성할 때에 전송 범위를 R1이라고 하면, R1= R/2로 설정하여 (4장에서 추가로 설명함), 동기화 단계에서 한 부모 노드의 자식 노드들은 전송 범위 R내에 있게 된다. Fig. 4는 제안한 EPSP의 동기화 알고리즘을 유사코드로 나타낸 것이다.

4. 시뮬레이션

알고리즘의 동작을 검증하기 위하여 NS2(Network Simulator)-2.35[12]을 사용하여 Fig. 5와 같이 무선 노드를 배치하고 실험하였다. 제안한 알고리즘을 사용하여 트리를 생성하였으며, 이 때 각 노드의 전송 전력을 다음의 동기화 동작 실험에서 사용할 전송 전력의 1/4로 설정하여 전송 범위를 약 1/2정도로 설정하였다.

Fig. 5에서 붉은 점선으로 표시된 원은 4번 노드를 루트 노드로 하여 트리를 구성할 때 설정된 전송 범위이다. 여기서 4번 노드(레벨 0)를 중심으로 노드 #0~#3, #5~#8이 자식 노드가 되고 트리의 다음 레벨인 1 레벨이 될 수 있다. 동기화 단계에서는 푸른 원으로 표시되는 전송 범위로 설정되어 레벨 1의 노드

```
<EPSP Synchronization Algorithm >
Use the Tree which was constructed from the
algorithm in Fig. 2.
Set Transmission Ranges to R_at all nodes
 For root node {
        send "INIT" with children list
  If (received all "REQ" from children)
        send "RES" with child_ID+time informations.
 }
 For child node {
  If (received "INIT" from parent && first child itself)
      send "REQ"
  If (received "REQ" from the previous cousin
heading the parent)
     send "REQ"
  If (received "RES" from parent)
      Do Synchronize
  If ( Parent node except root  ) { // Gateway
        send "INIT" with children list
 }
```

Fig. 4. Pseudo code of the proposed algorithm in Synchronization Phase.

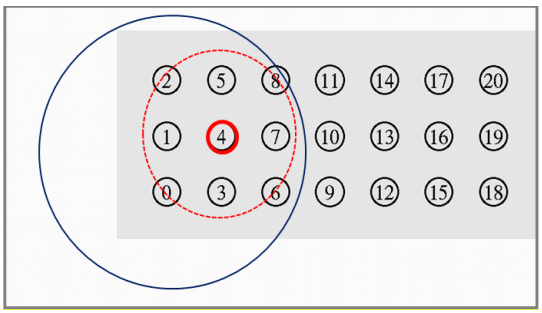

Fig. 5. Using different transmission ranges of nodes at two phases.

#0~#3, #5~#8이 전송 범위 내에 있게 되므로 상호 상대 노드를 overhearing할 수 있다. 동기화에서 동작하는 시뮬레이션 환경은 노드간 거리 100m, 전송 영역 반경 250m이고, 노드에서의 전력 변수로는 송신전력(0.660W), 수신전력(0.395W), 대기전력(0.035 W)으로 설정하였다. 사용된 MAC은 802.11 그리고 라우팅 프로토콜은 DSDV이다.

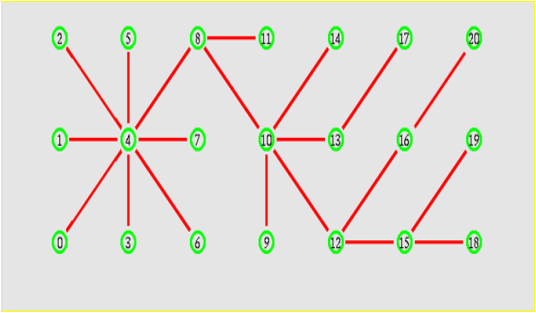

Fig. 6. Nodes deployment and tree constructed.

4.1 동기화 완료 시간

PSTN, P-TPSN, EPSP에 대하여 전체 노드의 동기화가 완료될 때까지 걸린 시간을 측정하였다(Fig. 7). PSTN에서는 각 노드가 부모 노드와의 패킷 전송을 시도할 때 이웃 노드와의 충돌을 피하기 위해 즉시 시도하지 않고 임의의 시간동안 기다린 후 전송을 시도한다. 그러므로 노드마다 다른 임의의 대기 시간에 의해 각 노드의 동기화 완료 시간이 결정된다. 그러므로 TPSN의 동작은 각 노드의 대기 시간을 (0~2.0 sec) 사이의 Uniform 함수에서 발생시켜 여러 번 시도한 것 동기화 완료 시간이 가장 빠른 것을 선택하였다.

P-TPSN과 EPSP는 TPSN과 달리 임의의 대기 시간을 사용하지 않으므로 결과는 항상 일정하다. Fig. 7은 3가지 방식의 동기화 완료 시간을 측정하고 비교한 것이다. EPSP이 가장 우수하고 다음 P-TPSN, TPSN순이다. 이는 Fig. 2에서 나타낸 것처럼, EPSP가 다른 두 방식보다 부모 노드와 자식 노드간의 데이터 교환 시 대기 시간이 짧기 때문이다. 다시 말하

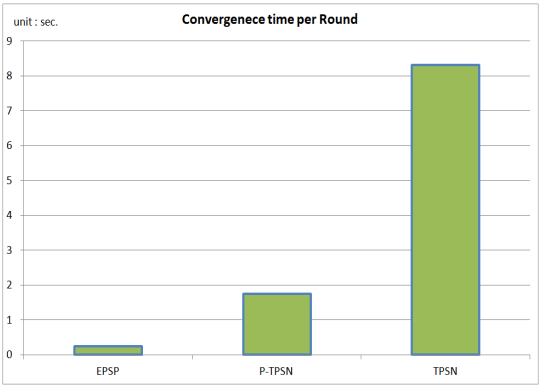

Fig. 7. Comparing Convergence times for 3 schemes.

면, P-TPSN은 TPSN에서 소요되는 임의의 대기 시간은 없지만, 부모 노드로부터의 지시를 기다리는 동안의 시간 지연이 필요하므로 TPSN보다는 우수하다. 여기에다 EPSP는 부모 노드로부터의 지시도 기다릴 필요 없이 전송을 시작하므로 P-TPSN보다 동기화 완료 시간이 더 우수하다.

3가지 방식에서 1 라운드를 동작했을 때 사용한 메시지 수를 측정하여 비교한 결과는 Fig. 8에 나타내었다. P-TPSN과 TPSN은 메시지 수에서는 차이가 없다 (즉, 동기화 완료 시간만 차이가 있다). 하지만, EPSP에서는 사용한 메시지 수가 줄어들었음을 알 수 있다(Fig. 3 참고). 전송 메시지 수의 변화는 전력 소모에도 영향을 끼칠 것임을 예상할 수 있다.

4.2 전력 소모

3가지 방식에 대하여 1 라운드 동안 동작시킨 후

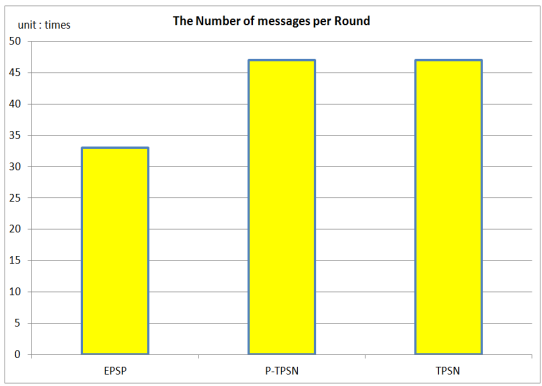

Fig. 8. Comparing the number of exchange messages per round for 3 schemes.

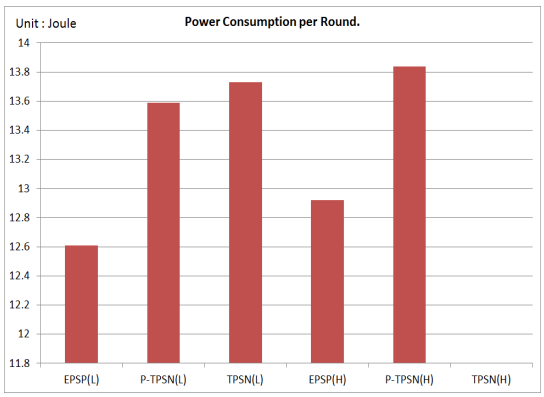

Fig. 9. Comparing Power-consumption per round for 3 schemes.

네트워크 전체의 소모 전력을 측정하여 결과를 Fig. 9에 나타내었다. 노드의 밀도에 따른 전력 소모를 측정하기 위하여 두 가지 토폴로지를 설정하였다. 노드 수는 동일하게 21개이고, 하나는 앞에서 설정한 노드 간격을 100m(그림에 L(Low density)로 표시)로 하고, 또 다른 하나는 노드 간격을 50m((그림에 H (High density)로 표시))로 하여 노드의 배치 밀도를 두 배로 하였다.

동일한 조건에서 충돌이나 대기 시간 등으로 인하여 TPSN이 가장 많은 전력을 소모하였고, P-TPSN은 충돌이나 대기 시간으로 인한 전력 소모가 TPSN보다 줄었다. 이와 비교하여 EPSP은 대기 시간이 가장 짧고 전송 메시지의 수가 적어 전력 소모가 가장 적게 나타났다.

EPSP와 P-TPSN이 노드 밀도가 높아질수록 전력 소모가 증가한 것은 생성된 트리에서 부모 노드들의 차수가 높아지므로 전송 영역 내에 있는 노드들의 수신 전력 소모가 증가한 것으로 보인다. 한편, TPSN은 저밀도에서 충돌이 발생하여 몇 번의 시도 후에 성공하였으나, 고밀도에서는 충돌이 지나치게 많이 발생하여 결과를 얻을 수 없었으며, 임의 대기 시간의 조정이 필요하다.

5. 결 론

TPSN은 무선 센서 네트워크에서 동작하는 대표적인 시각 동기화 프로토콜로서, 오늘날까지 많은 시각 동기화 프로토콜에서 활용되거나 참조가 되고 있다. 하지만, TPSN은 동기화 과정에서 여러 노드가 경쟁적으로 데이터 전송을 시도함으로써 충돌로 인하여 동작이 지연되거나 불필요한 전력이 소모된다. 그리고 TPSN에 경쟁방식 대신에 폴링 방식을 도입하여 불필요한 충돌을 줄임으로써, 전력 소모와 네트워크 전체 동기화 완료 시간을 줄인 P-TPSN이 제안되었다.

이 논문에서는 노드간의 데이터 전송 방식을 변화시켜 전송 메시지 수를 줄임으로써 전력 소모를 더욱 줄이고 동기화 완료 시간을 개선시킨 효율적인 시각 동기화 프로토콜인 EPSP를 제안하였다. 시뮬레이션을 사용하여 제안된 방안의 성능을 검증하고 기존의 다른 방안과 비교하여, 더 우수한 성능 결과를 확인하였다.

REFERENCE

[1] D. Mill, *Computer Network Time Synchronization: The Network Time Protocol on Earth and in Space*, Second Edition, CRC Press, 2011. New York, USA.

[2] S. Bae, "A Survey on Time Synchronization Protocols for Wireless Sensor Networks," *Journal of The Korea Society of Computer and Information*, Vol. 19, No. 6, pp. 61-69, 2014.

[3] S. Ganeriwal, R. Kumar, and M.B. Srivastava, "Timing-Synch Protocol for Sensor Networks," *Proceeding of ACM Sensys*, pp. 138-149, 2003.

[4] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The Flooding Time Synchronization Protocol," *Proceeding of ACM SenSys*, pp. 39-49, 2004.

[5] J Elson, L. Girod, and D. Estrin, "Fine-grained Network Time Synchronization Using Reference Broadcasts," *Proceeding of ACM OSDI*, pp. 147-163, 2002.

[6] S. Bae, "An EIBS Algorithm for Wireless Sensor Network with Life Time Prolongation," *Journal of the Korea Society of Computer and Information,* Vol. 19, No. 9, pp. 65-73, 2014.

[7] D. Liu, Z. Zheng, Z. Yuan, and W. Li, "An Improved TPSN Algorithm for Time Synchronization in Wireless Sensor Network," *Proceeding of International Conference on Distributed Computing Systems Workshops*, pp. 279-284, 2012.

[8] A. Kulaki and K. Erciyes, "Time Synchronization Algorithm Based on Timing-Synch Protocol in Wireless Sensor Networks," *Proceeding of International Symposium on Computer and Information Science*, pp.1-5, 2008.

[9] S. Bae, "A Revised Timing-sync Protocol for Sensor Networks by a Polling Method," *Journal of the Korea Society of Computer and Information,* Vol. 20, No. 8, pp. 23-28, 2015.

[10] K. Hur, "Realtime Clock Skew Estimator for Time Synchronization in Wireless Sensor Networks of WUSB and WBAN", *JOURNAL OF KOREA MULTIMEDIA SOCIETY* 15(11), pp. 1391-1398, 2012.

[11] S. Bae, "Enhanced Tree Construction Algorithm for Timing-sync Protocol for Sensor Networks," *Journal of Advances in Computer Networks*, Vol. 3, No.1, pp. 28-31, 2015.

[12] The Network Simulator ns-2: Documentation, http://www.isi.edu/nsnam/ns/ns-documentation.html, Nov. 2011. (accessed May, 2, 2016).

배 시 규

1998년 경북대학교 컴퓨터공학과 (박사)

1995년~현재 동양대학교 교수

관심분야 : Computer Networks, Mobile Networks, Internet Applications, IoT