제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

Bluetooth 장치 간의 홉 횟수를 고려한 트리 기반 scatternet 형성 알고리즘

강승호, 강대욱, 임형석
전남대학교 전산학과
e-mail: kinston@csblue.chonnam.ac.kr, {dwkang, hslim}@chonnam.chonnam.ac.kr

Scatternet Formation Algorithm Based on Tree Topology Considering Hop Count Between Bluetooth Devices

Seung-Ho Kang, Dae-Wook Kang, Hyeong-Seok Lim
Dept of Computer Science, Chonnam University

요 약

다수의 Bluetooth 장치가 사용되는 곳에서의 데이터 통신에는 scatternet의 형태가 데이터 전송 효율성에 크게 영향을 미친다. 본 논문에서는 과도한 지연시간을 초래하지 않으면서도 장치 간 평균 홉 횟수를 기존 방법보다 감소시킬 수 있는 scatternet 형성 알고리즘을 제시하고 시뮬레이션을 통해 기존 알고리즘과 비교한다.

1. 서론

Bluetooth는 원래 이동전화, 헤드셋, 노트북과 같은 장치들을 연결하고 있는 케이블을 대체할 목적으로 개발된 저 비용, 저 전력, 근거리 무선통신기술이었다[2,3]. 그러나 저 비용, 저 전력 등의 장점이 이 기술의 범용성을 증가시킴에 따라 단지 케이블의 대체 기술에만 머무는 것이 아니라 PAN(Personal Area Network)의 개념을 탄생시키는[3] 등 기존 ad hoc 네트워킹 기술의 여러 측면을 진일보시켜 나가는 새로운 무선통신의 표준으로 자리 잡아가고 있다.

Bluetooth는 2.4GHz ISM 주파수대에서 주파수 홉핑 기술을 채택하고 기존의 무선 LAN과는 다른 마스터-슬레이브 매커니즘에 기반한 MAC 프로토콜을 사용한다. 이는 ISM 주파수대에서 사용되고 있는 여러 다른 장치들이나 Bluetooth 장치 상호간의 통신상의 간섭을 감소시키고 보안상의 장점을 제공한다. 통신상 간섭의 감소는 통신이 가능한 일정한 영역 안에 다수의 Bluetooth 장치간의 상호 통신의 가능성을 높임과 더불어 이들 간의 효율적인 통신 방법 또한 요구하게 되었다. 이러한 요구는 하나의 piconet 내부에서만이 아니라 이들 piconet들 간의 네크워크를 형성하는 방법, 즉 scatternet의 효율적인 형성 방법까지 확대되었으며 이에 대해 여러 방법들이 제시되고 있다. 그중 1) scatternet의 링크 수 최소화 2) bridge 노드의 참여 piconet 수 최소화 3) 노드 간 경로의 유일성 등의 장점에 근거한 트리 형태의 형성방법이 주목을 받고 있다[6,8].

본 논문에서는 이러한 트리의 형성과정에 자식노드 수를 고려함으로써 결과적으로 얻어진 scatternet 상의 장치들 간의 평균 홉 횟수를 감소시켜 통신상의 효율성을 증대시키는 방법을 제시한다.

2. 관련 연구

현재 scatternet의 형성에 관한 연구는 ad hoc 네트워크에서 완전한 연결성의 보장, 형성과정에서의 최소의 지연시간 경과, 그리고 결과적으로 데이터 전송상의 효율성을 증진시키는 방법을 찾는 데 모아지고 있다. 이에 관해 현재 제시되고 있는 방법들로는 Bluetree[8], BTCP[5], TSF[6,7]등이 있다.

Bluetree는 일단 Inquiry단계가 끝난 상태에서 임의의 장치가 root가 되어 spanning tree형태로

scatternet을 형성해 가는 방법이다. 이 방법은 트리 토폴로지가 갖는 장점을 가지고 있으나 실제 형성 상 지연시간의 대부분은 Inquiry단계에서 발생하는데도 이 점이 무시되고 있고 장치의 상태가 네트워크 진입 시에 고정되어 있음을 가정하는 등이 단점으로 지적된다.

BTCP(Bluetooth Topology Construction Protocol)는 상호통신이 가능한 일정한 장소에 일정한 수의 장치가 존재하는 경우 scatternet을 주관할 조정자를 선정하고 이 조정자가 전체 scatternet을 형성해 가는 방법이다. 선정된 조정자는 전체 장치 수를 고려하여 필요한 최소수의 마스터들을 선정하고 그 마스터들을 대상으로 piconet을 형성한 후 이들 마스터가 각각 나머지 디바이스들에 대해 piconet을 형성하도록 하여 전체 scatternet을 형성하는 것이다. 이 방법은 네크워크 상에 존재하는 장치들이 진입 시 부여된 고정된 역할을 수행하는 것이 아니라 임의의 시간 간격으로 inquiry와 inquiry scan을 번갈아 가며 수행한다고 가정하여 현실성을 높였다는 점과 36개 이하의 장치 수에 대해서 최소의 piconet 수를 제시할 수 있다는 점이 주목할 만하지만 디바이스 수가 36을 초과한 경우에는 적절한 방법을 제시하지 못하고 있다는 한계가 있다[6].

그리고 본 논문은 TSF의 기본적인 가정을 수용하고 scatternet 형성 과정상에 새로운 아이디어를 첨가하여 이를 개선코자 한 것으로, 다음 절에서 TSF 방법을 보다 구체적으로 살펴보고 새로운 알고리즘을 제시하였다.

3. TSF (Tree Scatternet Formation)

TSF는 BTCP가 제시한 노드의 inquiry 상태와 inquiry scan 상태의 전이 가정을 기반으로 하고 있다. 여기에 각 노드를 트리의 형성 과정에서의 지위에 따라 Root 노드, Non-root 노드, 그리고 Free 노드로 구분하여 역할을 부여한다. 일단 모든 노드는 Free 노드의 지위에서 트리 형성을 시도하게 되고 일단 노드 간 회합이 있게 되면 inquiry를 수행하는 쪽은 Root 노드의 지위를, inquiry scan을 수행하는 쪽은 Non-root 노드의 지위를 갖게 된다. 그리고 트리 형태의 scatternet을 형성키 위해 다음과 같은 세 가지 규칙[6]에 따라 형성이 진행된다.

1) Free 노드는 다른 Free 노드 또는 Non-root 노드하고만 연결을 시도한다.
2) Root 노드는 오직 다른 Root 노드하고만 연결을 시도한다.
3) Non-root 노드는 오직 Free 노드하고만 연결을 시도한다.

이때 Non-root 노드는 inquiry 상태에만 머물고 Free 노드와 Root 노드는 random(E[T_{inq}],D)의 간격으로 inquiry 상태와 inquiry scan 상태를 번갈아 가며 트리 형성상의 역할을 수행한다.

E[T_{inq}]는 Inquiry 단계를 수행하는 데 걸릴 것으로 예상되는 시간으로 시뮬레이션 결과 1.8s정도이며 D는 임의 간격의 크기로서 $2.2 \le D \le 4.4$ 라고 알려져 있다[7]. 따라서 실제 한 노드가 inquiry 또는 inquiry scan 상태에 머무는 시간은 $2.2 * E[T_{inq}] \le random(E[T_{inq}],D) \le 4.4 * E[T_{inq}]$ 인 것이다.

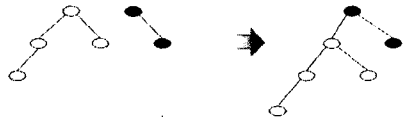
이 방법은 BTCP가 제시한 역할 전이라는 현실성 있는 가정을 채택하고 노드 수에 관계없이 트리 형태의 scatternet 형성이 가능하다는 장점을 가지고 있다. 그러나 이 방법은 Inquiry단계에서 각 노드가 임의적으로 주어진 시간만을 *inquiry*나 *inquiry* scan에 소모한다고 가정함으로써 결과적으로 얻어질 scatternet의 형태가 전체 데이터 전송상의 효율성에 중요한 영향을 미친다는 사실을 적절히 반영하지 못하고 있다.

4. 제안 알고리즘

일반적으로 같은 노드수의 트리라면 높이가 낮은 트리의 노드 간 평균거리가 보다 짧은 것으로 알려져 있다. 본 논문이 제시하는 알고리즘은 이러한 트리의 특성에 착안하여 scatternet을 형성하는 과정에 일정한 방법을 적용하여 높이가 낮은 트리 형태의 scatternet을 형성해 냄으로써 장치 간 홉 횟수를 줄이고 결국 데이터 통신의 효율성을 높이고자 하는 것이다.

여러 트리로 구성되어 있는 포레스트를 하나의 트리로 형성해 나갈 때 전체적으로 높이가 높지 않도록 하는 방법은 (그림 4-1)의 (a)처럼 높이가 높은 트리의 루트가 높이가 작은 트리의 루트를 자식 노드로 삼도록 하는 것이다.

(a)



(b)

(그림 4-1)

이 방법을 TSF의 Root 노드에 적절히 반영하면 형성되는 scatternet의 높이가 낮아지고 전체 노드 간 홉 횟수를 줄일 수 있을 것이며 따라고 데이터 통신의 효율성도 개선할 수 있을 것이다.

그러나 이 방법을 Inquiry 단계에서의 scatternet 형성과정에 그대로 적용할 수는 없다. 현재 inquiry 단계에서 사용되는 IAC(Inquiry Access Code) 패킷이나 FHS(Frequency Hop Synchronization) 패킷에는 자신이 속한 트리의 높이나 노드 수를 담아서 보낼 공간이 할당되어 있지 않기 때문이다.

이러한 명세서상의 제약사항을 만족시키면서 본 논문이 제시하는 방법은 다음과 같다.

Free 노드의 지위로부터 inquiry를 수행하여 최초로 Root 노드가 된 때부터 각 Root 노드는 자신이 inquiry를 수행하는 중 얻어진 FHS 패킷 수를 고려하여 그 패킷수에 비례하는 일정 시간 T_{inq}([식 4-1])을 inquiry를 수행하고 패킷수에 반비례하는 일정시간 $T_{i\text{-}scan}$([식 4-2])은 inquiry scan을 수행하도록 하는 것이다. 이 방법은 결국 inquiry를 수행하며 모은 FHS 패킷 수가 많은 쪽의 Root 노드가 적은 쪽의 Root 노드를 scatternet 형성 시에 자식노드로 삼자는 것이다.

$$T_{inq} = N_{FHS} * E(T_{inq}) \quad [식\ 4\text{-}1]$$

$$T_{i\text{-}scan} = (7 - N_{FHS}) * E(T_{inq}) \quad [식\ 4\text{-}2]$$

그리고 나머지 Free 노드나 Non-root 노드는 TSF에서 제시한 것과 같은 시간 간격의 역할 패턴을 따르는 것으로 하였다.

아래에 각 노드의 상태-기계를 의사코드로 제시한다.

```
PROCEDURE FREE() {
   do {
      state <--- OPPOSITE(state)
      t_state <--- random(E[Tinq],D)
      현재 상태에서 t_state 동안 머무르며 역할
      수행
   } while (!FHS())

PROCEDURE ROOT() {
   A: do {
      state <--- OPPOSITE(state)
      if (state = inquiry)
         t_state <--- NFHS * E(Tinq)
      else if (state = inquiry scan)
         t_state <--- (7 - NFHS) * E(Tinq)
      t_state 동안 각 상태에서 역할 수행
   } while(!FHS())
   if (state = inquiry) goto A
   else Non-root 노드로 지위 변화
}
PROCEDURE NON-ROOT() {
   state <--- inquiry
   계속해서 inquiry 상태 유지
}
PROCEDURE FHS() {
   if ( send or receive FHS packet)
        return 1
   else  return 0
}
PROCEDURE OPPOSITE(state) {
   if (state = inquiry)
        state <--- inquiry scan
   else
        state <--- inquiry
   return state
}
```

이 방법이 갖는 타당성의 근거는 다음과 같다. Root 노드는 오직 Root 노드하고만 연결을 시도할 수 있는데(규칙 2)) FHS 패킷을 많이 보유하고 있다는 것은 그동안 그만큼 많은 수의 다른 트리들을 자신의 하위에 두었다는 것이고 그만큼 트리의 높이나 노드 수면에서 성장했다는 것을 의미한다. 따라서 FHS 패킷 수에 비례하게 inquiry 기간을 늘여고 반대로 inquiry scan 기간을 줄여주면 그만큼 다른 작은 트리의 Root 노드를 자식으로 삼을 상대적인 기회를 늘려 줄 수 있다.

5. 시뮬레이션 및 성능 비교

본 논문에서 제시한 알고리즘은 ns-2[4]와 IBM에서 제공한 BlueHoc 모듈[1] 부분을 수정하여 시뮬레이션 하였다.

모든 노드는 20 x 20m의 장소에 랜덤하게 위치하게 하였으며 15초 이내에 랜덤하게 진입하는 것으로 하였다. 진입 시 Free노드의 상태는 inquiry와 inquiry scan 상태가 반씩 구성되도록 하였다. 또한 TSF와의 공정한 비교를 위하여 $E(T_{inq})$나 D값은 TSF의 것을 그대로 사용하였다.

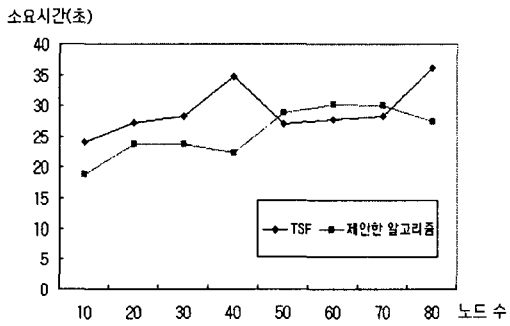
시뮬레이션은 inquriy scan 상태에서 실제 scan을 수행하는 기간인 scan window 값으로 명세서가 권장하는 18슬롯(11.25ms)으로 하고 실행하였다. scan window의 값은 inquiry scan 상태의 기간과 동일한 시간을 가질 수 있지만 전력소모의 측면이나

이미 네트워크에 연결된 상태의 노드라면 수행해야 할 다른 일이 있을 수 있기 때문에 이와 같은 시간이 권장되고 있는 것이다.

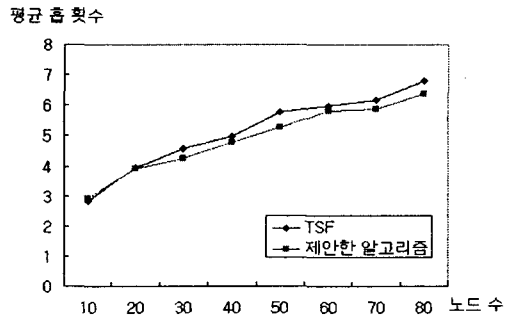
그리고 그래프상의 각 점은 각각의 방법에 따라 10번씩 수행한 후 평균치로 나타내었다.

(그림 5-1)에서 보듯이 전체적인 scatternet이 형성되기까지 소요되는 시간은 TSF 나 본 논문이 제안한 알고리즘이나 별다른 차이가 없었다. 또한 노드수의 증가가 scatternet 형성에 소요되는 시간에 미치는 영향도 그다지 크지 않은 것으로 나타났다. 아마도 각 노드의 연결이 병렬 적으로 발생하기 때문일 것이다.

노드 간 평균 홉 횟수는 (그림 5-2)에 나타나듯이 노드 수가 적은 경우에는 별다른 차이가 보이지 않았으나 노드 수가 증가 할수록 본 논문이 제시한 알고리즘에 의해 형성된 scatternet이 적어지는 경향을 보였다.



(그림 5-1)



(그림 5-2)

6. 결론 및 향후 연구과제

본 논문은 같은 수의 노드라면 높이가 낮은 트리의 노드 간 평균 거리가 짧다는 트리의 특성을 이용하고 Bluetooth 명세서 상의 제약 사항을 만족하면서 효율적인 scattenet을 형성하는 알고리즘을 제시하였다. 실제, 장치들이 많고 교환되는 데이터 양이 많다면 scattenet 형태에 의해 좌우되는 홉 횟수는 전체적인 네트워크의 효율성에 큰 영향을 미치게 될 것이다.

따라서 향후 연구방향은 장기적으로 그래프 이론 등이 제시하는 여러 토폴로지의 장단점을 검토하여 scatternet의 형성에 이용하는 것이고, 본 논문과 관련하여서는 Root노드의 자식노드 수를 적절히 반영하는 함수를 개발해 내는 것이다. 아울러 Free 노드나 Non-root 노드의 역할도 효율적인 scatternet을 구성하데 일조 할 수 있는 방법을 찾는 것이다.

참고: 이 논문은 2001년도 전남대학교 학술연구비 지원에 의하여 연구되었음.

참고문헌

[1] "Bluetooth Extension for ns," http://oss.software.ibm.com/developerworks/opensource/bluehoc/, 2001.

[2] Bluetooth SIG, "Specification of the Bluetooth System ver 1.1," http://www.bluetooth.com/ February 2001.

[3] J. Bray and C. F. Sturman, *BLUETOOTH 1.1 Connect Without Cables*, 2nd Ed, Prentice Hall, 2001.

[4] "ns-2 Network Simulator," http://www.isi.e여 /nsnam/vint/, 2000.

[5] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. LaMaire, "Distributed Topology Construction of Bluetooth Personal Area Networks," Proc. IEEE INFOCOM, April 2001.

[6] G. Tan, A. Miu, J. Guttag, and H. Balakrishnan, "Forming Scatternets from Bluetooth Personal Area Networks," MIT-LCS-TR-826, October 2001.

[7] G. Tan, "Self-organizing Bluetooth Scatternets," MS Thesis, MIT, January 2002.

[8] G. V. Zaruba, S. Basagni, and I. Chlamtac, "Bluetrees-Scatternet Formation to Enable Bluetooth-Based Ad Hoc Networks," ICC 2001, vol.1, pp. 273-277, June 2001.