Density를 고려한 센서 네트워크 Reprogramming 프로토콜

조성규° 차호정
연세대학교 컴퓨터과학과
{skcho°, hjcha}@cs.yonsei.ac.kr

Density-Adaptive Sensor Network Reprogramming Protocol

SungKew Cho° Hojung Cha
Dept. of Computer Science, Yonsei University

요 약

Network Reprogramming 에서는 코드 전파가 완료되기까지 걸리는 시간과 에너지 소모가 문제가 된다. 지금까지의 연구들은 에너지 소모를 줄이기 위해서 완료시간을 줄이는 방법에 초점을 두었지만 제한된 에너지를 사용해야 하는 센서 네트워크에서는 전송된 메시지에 의한 에너지 소모도 고려해야 한다. 본 논문에서는 코드 전파 시 전송 메시지 개수를 최소로 하는 DANP를 제시한다. DANP는 각 노드가 이웃 노드에 대한 정보를 이용하여 advertisement 주기를 적절히 늘려서 모든 경우의 Density에 대해서 코드 전파의 효율성을 높인다.

1. 서 론

센서네트워크는 주로 환경 모니터링이나 위치 추적과 같은 장기간의 정보를 필요로 하는 곳에 사용되며 이 때문에 여러 개의 센서 노드가 배치된 후에 환경의 변화나 원하는 정보의 변화 때문에 각 센서 노드에서 실행되는 프로그램을 수정하거나 바꿔야 할 경우가 있다. 이미 배치된 센서 노드의 프로그램을 일일이 하나씩 바꾸는 것은 어려우므로 이 때 *network programming* 이 필요하다.

센서 네트워크에서 코드전파 방법에 관한 대표적인 연구로는 XNP[1], MOAP[2], Deluge[3], GARUDA[4], MNP[5]등이 있다. XNP는 base station이 자신과 통신할 수 있는 범위 안의 노드들에게 코드 캡슐을 주는 방법을 사용한다. 이 방법은 멀티 홉을 통한 코드 전파를 할 수가 없다는 단점이 있다. MOAP의 코드 전파는 새로운 코드를 가진 노드가 publish를 하면 새로운 코드를 원하는 노드가 subscribe를 하는 방식을 사용한다. 이를 통해 멀티 홉에서의 코드 전파가 가능하다. Deluge는 MOAP와 유사한 방법을 사용하지만 MOAP와 달리 이미지를 분할하여 전송하며 이로 인해 전체 코드의 일부만 가진 노드가 소스가 될 수 있어서 코드 전파를 완료하는데 까지 걸리는 시간이 감소하게 된다. GARUDA는 WFP(wait for first packet)라는 강한 신호를 이용하여 코드 전파전에 flooding 을 통해 자신에게 코드를 줄 소스를 정함으로써 신뢰할 수 있는 코드전파를 가능하게 한다. MNP는 어떤 노드를 기준으로 했을 때 그 노드의 이웃들 중 오직 한 개만 data를 보낼 수 있도록 함으로써(sender selection) Hidden Terminal Problem[6]을 해결 하려고 하였으며 자신이 선택되지 않을 경우 radio를 off 함으로써 idle listening time 을 줄인다.

기존의 코드전파에 관한 연구들은 에너지 소모에 가장 큰 척도가 되는 코드 전파 완료 시간을 줄이는 것을 목표로 하며 Hidden Terminal Problem을 큰 문제라고 본다. 이를 해결하기 위해 MNP는 ADV/REQ를 RTS/CTS 대신 사용하는 방법을 제시한다. 하지만 코드 전파와 같은 1 to n 통신의 경우 CTS를 대신하는 REQ 메시지를 ADV 메시지를 들은 노드 전체가 전송해야만 1 to n 통신의 장점을 살릴 수 있다. 이로 인해 메시지 overhead가 크다. GARUDA는 미리 sender와 receiver를 정해 놓고 sender들이 동시에 데이터를 보내도 HTP가 발생하지 않도록 하는 방법이 있다. 하지만 이 방법은 특별한 H/W를 통해 WFP라는 강한 신호를 이용해야만 가능하다는 단점이 있다. Deluge는 코드의 일부를 받으면 얼마간 코드의 다른 부분을 요구하지 못하도록 하는 시간 지연을 줘서 HTP를 줄인다. DANP는 기존 방법들과는 다르게 코드 전파 완료 시간에 영향은 없으면서 총 전송 메시지 개수를 줄이려는 데 목적이 있다. 이것을 성취하기 위해 각 노드는 이웃 노드 정보를 이용하여 그에 적합한 코드 전파 정책을 정한다. 이러한 이유로 다양한 density 에 대해 좋은 성능을 보일 수 있다.

본 연구는 과학기술부에서 지원하는 국가지정연구실사업으로 수행하였음 (과제번호 : 2005-01352)

2. 센서 네트워크 Reprogramming

이번 절에서는 기존에 제시된 Network Reprogramming Protocol 중 대표적인 방법인 Deluge에 대해 살펴보고 이것의 문제점을 알아본다. Deluge는 data를 전파하기 위해서 Advertisement-Request-Data의 3-way protocol을 사용한다. 모든 노드들은 Advertisement 메시지에 자신이 현재 갖고 있는 data에 대한 정보를 담아서 주기적으로 전송한다. 만약 어떤 노드가 수신한 advertisement 메시지를 통해서 새로운 data가 있다는 것을 알게 되면 송신한 노드에게 Request 메시지를 보내고 Request를 받은 노드는 새로운 Data를 송신한다. Deluge는 density에 관계없이 최소한의 Advertisement메시지 전송으로 code 정보를 유지, 관리하기 위해 suppression을 사용하며 이를 빠르고 효과적으로 하기 위해 advertisement 주기를 상황에 맞게 늘리고 줄인다. suppression이란 network congestion을 줄이기 위해서 자신이 송신하려는 메시지와 같은 내용의 메시지를 엿듣게 되면 자신의 메시지를 얼마 후에 보내거나 보내지 않는 것을 뜻하며 이를 사용하면 모든 노드의 advertisement 주기가 t라고 할 때 일반적으로 어떤 cell에서 시간 t동안 advertisement 메시지를 전송하는 노드는 두 개다. Deluge는 selective NACK를 사용하여 재전송을 요구한다. 일반적으로 코드는 n개의 패킷으로 구성되며 n개의 패킷에 대해 n bit를 사용하여 받지 못한 패킷의 bit를 set하여 Request를 보낸다. Request 메시지는 unicast 방식을 사용하며 DATA 메시지는 broadcast 방식을 사용한다. Deluge는 전체 코드를 몇 개의 page로 나눈 후 pipelining을 이용하여 서로 다른 page를 동시

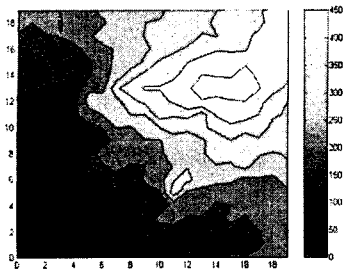

그림 1. 20x20, 10feet spacing에서 1개의 page전파 시 각 노드별 전파 완료 시각

에 전송한다.

Delgue의 문제는 dense한 환경에서 코드 전파 시 가운데 부분 노드들의 코드 전파 완료 시간이 edge 노드들의 완료 시간보다 훨씬 긴 현상이 발생하는 것이다. 이 문제의 원인은 Deluge 논문에서 잘 지적했듯이 suppression을 실패하는 데에 있다. Deluge는 suppression을 이용하여 통신량을 조절하는 데 suppression은 메시지를 잘 들을 수 있는 상황에서 가능하다. 코드 전파를 하면 통신량이 많아지고 이로 인해서 HTP이 많이 발생하여 suppression을 실패하게 된다. 이것은 통신량을 증가시켜 다시 suppression을 실패하는 원인이 되는 악순환이 된다. 그림 1은 Deluge를 사용하여 1개의 page를 20x20 10feet spacing에서 전파 시 완료 시각을 보여준다. edge 부분의 노드들 보다 가운데 부분의 노드들의 완료 시간이 더 오래 걸리는 것을 알 수 있다.

3. Density-Adaptive Network Reprogramming Protocol

DANP는 dense한 환경에서 suppression 방법의 한계를 해결하기 위해서 모든 메시지의 발단이 되는 ADV 메시지의 주기(Deluge에서 최소 ADV 주기)를 조정한다. 이웃 노드 개수가 많아질수록 ADV 메시지의 주기를 늘리면 자체적으로 메시지 개수를 조절하게 되므로 density가 커질수록 HTP가 커지는 문제를 해결할 수 있다. 그러나 ADV 메시지 주기가 커지면 ADV 메시지를 받기까지 시간이 오래 걸려서 코드 전파 완료 시간이 길어질 수 있다. 그러므로 ADV 주기를 늘려도 코드 전파 완료 시간이 커지지 않는다는 근거가 필요하다. density가 큰 환경(이웃 노드 개수가 많은 환경)에서 ADV 메시지 주기를 증가시킬 수 있는 근거는 다음과 같다. 그림 2의 a에서 A가 한 번 ADV 메시지를 전송하면 1개의 REQ 메시지가 전송된다. 반면 b의 경우 A가 한 번 ADV 메시지를 전송하면 4개의 REQ 메시지가 전송된다. REQ가 많다는 것은 처리해야 할 일이 많다는 것이므로 b에서 노드 A의 처리시간이 a에서 노드 A의 처리시간보다 오래 걸린다. b에서 노드 B, C, D, E는 A가 자신들이 보낸 REQ에 대해 DATA를 보내는 동안 아무 일도 하지 못한다. 그럼에도 Deluge를 사용하고 만약 ADV 주기 중 최소 주기 t의 반이 A가 4개의 REQ를 처리하는 시간보다 짧다면 Deluge 방법에 의해 B, C, D, E는 A로부터 DATA를 받는 동안 F 혹은 G로부터 ADV 메시지를 받게 된다. 이러한 메시지는 불필요하므로 ADV 주기 t를 A가 4 개의 REQ를 처리하는 시간의 두 배로 정할 수 있다. 그러므로 일반적인 ADV 주기 t는 2×예상처리시간 이다. ADV를 보낸 후 3개의 이웃노드의 REQ를 처리하는 과정은 그림 2의 (c)와 같으며 한 개의 이웃 노드의 REQ를 처리하는 데에는 REQ를 보내기까지의 시간(REQdelay)와 DATA를 보내기까지의 시간(DATAdelay)가 필요하다. 일반적으로 DATAdelay값은 0이다. 예상처리시간을 구

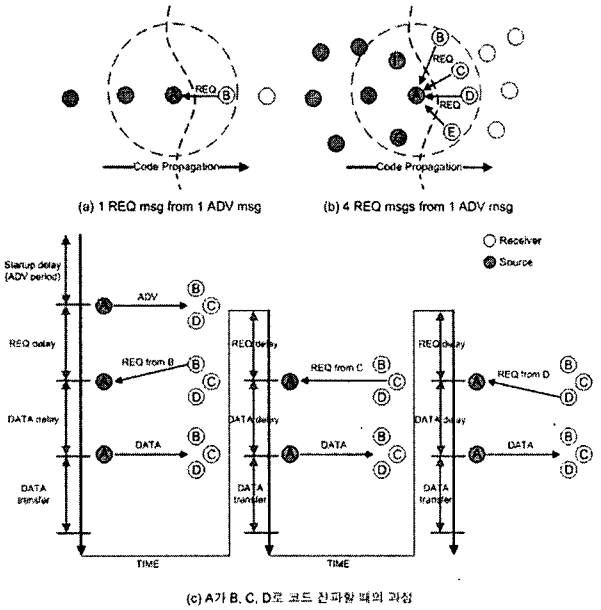

그림 2. 1개의 ADV 메시지에 대한 REQ 메시지의 처리

하는 것은 복잡하므로 다음과 같이 단순화한다. 어떤 노드 A의 이웃 노드 n개 중 kn개(0□□k□□1)는 A가 새로운 page에 대한 ADV 메시지를 보내면 page 전체를 요구한다고 가정한다. 이제 A가 새로운 page에 대한 ADV 메시지를 보내면 kn개의 이웃노드 중 한 개가 page전체를 요구할 것이고 A는 page에 대한 모든 DATA를 전송할 것이다. 이제 kn개의 이웃 노드들은 loss rate에 의해 A가 보낸 page 중 일부를 받게 되고 받지 못한 부분에 대해서 REQ 메시지를 보낼 것이다.(re-transmission) DATA를 보낼 확률은 ADV 메시지를 받을 확률과 REQ 메시지를 받을 확률의 곱이므로 ADV메시지를 보내는 노드의 입장에서 보면 이웃노드 i에게 DATA를 보낼 확률은 upLinkQ(i)×downLinkQ(i)이다. 이것을 정리해 보면 다음과 같다.

$$ADVperiod/2 = bestUpDownLinkQ \times (REQdelay + \max DataTxTime) + reTXTimeForEachNeighbor \times k \quad (1)$$

bestUpDownLinkQ는 모든 이웃 노드 i에 대한 upLinkQ(i)×downLinkQ(i) 중 최대값이고 maxDataTxTime은 한 개의 page 전체를 전송할 때 걸리는 시간이다. bestUpDownLinkQ × (REQdelay + maxDataTxTime)은 앞서 단순화한 것에서 page전체를 요구한 것을 처리하는데 걸리는 시간이며 reTXTimeForEachNeighbor은 이웃 노드 각각에 대해 재전송을 처리하는데 걸리는 시간이다. 재전송 처리 시간은 다음과 같이 구할 수 있다.

$$reTXTimeForEachNeighbor = \sum_{i=1}^{n}[upLinkQ(i) \times downLinkQ(i) \times (REQdelay + (1 - upLinkQ(i)) \times \max DataTXtime)] \quad (2)$$

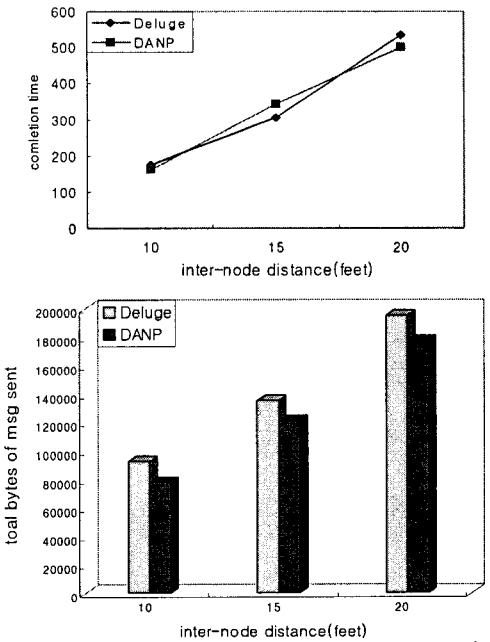

그림 3. 10x10환경에서 1 개의 page(1KB)를 전파 할 때 완료시간과 총 전송된 메시지의 byte 크기

이제 이웃 노드 환경에 적절한 ADV 주기를 구하였다. 그러나 이 값을 그대로 사용하면 코드 전파가 느려진다. 왜냐하면 이 값은 대부분의 kn개의 이웃노드들이 새로운 page의 데이터를 갖고 있지 않으며 또한 한 번의 새로운 page에 대한 ADV 메시지에 대부분의 kn개 이웃노드들이 REQ를 보낸다는 조건이 전제되기 때문이다. 만약 이러한 조건을 만족하지 않는다면 한 번의 ADV 메시지에 대한 처리 시간은 짧아진다. 처리 시간이 짧아졌기 때문에 ADV 메시지에 대해서 REQ 메시지를 보낼 수 있지만 ADV 주기를 크게 정하였으므로 얼마간 ADV 메시지를 받지 못하고 결과적으로 코드를 받기까지 시간이 길어진다. 그림 2의 c에서 ADV 메시지를 받기까지 걸리는 시간인 startup delay가 이를 나타낸다. 코드 전파를 빨리 하기 위해서는 ADV 메시지도 빨리 받아야 한다. ADV 메시지 주기를 길게 정한 상황에서 이러한 문제를 해결하기 위해서 새로운 page를 모두 받은 노드들은 반드시 몇 번은 ADV 주기를 작은 값으로 설정한다.

4. 성능평가

Deluge와 DANP의 성능을 비교하기 위하여 센서네트워크에서 많이 쓰이는 시뮬레이터인 TOSSIM을 사용하였고 Lossy Builder를 사용하여 노드 간 loss rate을 정하였다. 실험은 10x10의 grid 환경에서 노드간 거리를 10, 15, 20 feet의 3가지 경우와 20x20, 10feet spacing의 환경에서 진행하였다. 각 노드가 이웃 노드 정보를 알고 있어야 하므로 코드 전파전에 5초 간격으로 20회 자신의 id를 담은 패킷을 방송 후 ADV 주기를 정하고 1KB의 코드(1개의 페이지)를 그리드 배치의 모서리(0, 0) 노드에서 전파하기 시작하여 모든 노드가 코드를 다 받을 때까지 실험하였다. 코드를 다 받으면 2번은 ADV 주기를 2초로 하여 startup delay를 줄였고 앞서 제시한 이웃 노드 중 receiver가 될 확률 k는 1/2로 정하였다. 그림 3은 10x10환경에서 3가지 spacing에 대해 1개의 page를 전파할 때 총 전송

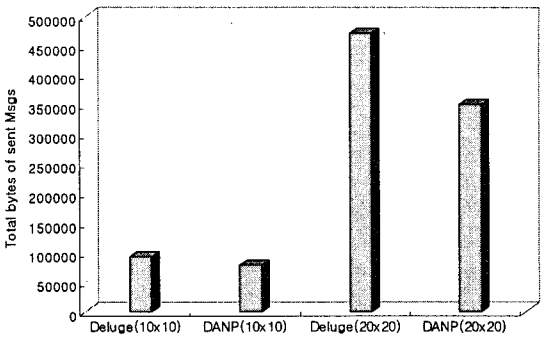

그림 4. 10x10, 20x20 환경에서 노드간 간격이 10feet 일때 Deluge와 DANP 의 총 전송 byte 크기

byte와 완료 시간을 측정한 것이다. 실험 결과 DANP는 3가지 density에 대해서 메시지를 Deluge보다 적게 보냈고 완료시간은 Deluge와 비슷하여 ADV 주기를 늘렸다고 완료시간이 길어지지 않음을 확인할 수 있었다. 그림 4는 10x10과 20x20 환경에서 코드 전파가 완료되기까지 Deluge와 DANP의 전송 메시지를 byte로 환산한 값이다. 결과에서 알 수 있듯이 20x20일 경우 10x10보다 HTP이 많이 발생하므로 ADV주기를 늘리면 에너지 절감에 의한 이득이 더 크고 이는 약 26%이다.

5. 결 론

본 논문에서는 에너지 효율적인 코드 전파를 위해서 코드 전파 완료 시간에 영향은 없으면서 총 전송 메시지 수를 줄이는 방법을 기술하였다. 이것은 이웃 노드 정보를 이용하여 ADV 주기를 변화시키는 것이다. 1 to n 통신의 특성을 갖는 코드 전파에서는 density가 커지면 n이 커져서 소스노드의 작업량이 증가한다. 이를 이용하여 ADV 주기를 늘릴 수 있다. 그러나 주기를 늘리면 startup delay가 발생하며 이를 없애기 위해서 코드 전파를 할 때 코드를 새로 받으면 반드시 몇 번은 ADV 주기를 짧게 해야 한다. 실험 결과 density가 커질수록 DANP 가 deluge 보다 총 전송 메시지는 적으면서 코드전파 완료 시간은 비슷한 것을 볼 수 있다.

참고문헌

[1] Crossbow Tehnology Inc. Mote in-network programming user reference, http://webs.cs.berkeley.edu/tos/tinyos-1.x/doc/xnp.pdf.

[2] T. Stathopoulos, J. Heidemann, and D. Estrin, "A remote code update mechanism for wireless sensor networks", Technical report, UCLA, Los Angeles, CA, USA, 2003.

[3] Jonathan W. Hui, David Culler, "The Dynamic Behavior of a Data Dissemination Protocol for Network Programming at Scale", *In the Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys'04), pages 81~94, publication 2004.*

[4] Seung-Jong Park, Ramanuja Vedantham, Raghupathy Sivakumar and Ian F. Akyiliz, "A Scalable Approach for Reliable Downstream Data Delivery in WSN", *In Proceedings of the 5th ACM International Symposium on Mobile Ad hoc Networking and Computing (MOBIHOC)*, May 2004.

[5] S.S.Kulkarni and Limin Wang, "MNP: Multihop Network Reprogramming Service for Sensor Networks", to appear in the 25th International Conference on Deistributed Computing Systems(ICDCS). June 2005. Columbus, OH.

[6] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," Wireless Networks, vol. 8, no. 2-3, pp. 153-167, 2002.