AUCPD : USN기반 패턴 데이터에 대한 적응적인 압축 알고리즘

정성민* 조인휘* 송병훈**
한양대학교 전자컴퓨터통신공학과* 한국 전자부품연구원**
{smjung, iwjoe}@hanyang.ac.kr
bhsong@keti.re.kr**

Adaptive Ultra-Compact Algorithm for Pattern Data based on USN

Sungmin Jung* Inwhee Joe* Byounghun Song**
Department of Electronics and computer engineering Hanyang university*, Korea Korean Electronics Technology Institute**

요 약

현재 센서 네트워크기반에 다양한 적용으로 인하여 데이터 통신에 량이 많아지고 있다. 기존에 환경 모니터링 등과 같이 조도, 온도, 습도를 다루는 것에서 ECG, EKG, GPS등과 같은 비교적 센서 네트워크 환경에서 대용량 데이터를 다루어지고 있다. 이러한 점은 희박한 자원을 바탕하는 USN환경에서 문제가 된다. 이 문제는 기존 연구 방향에서 데이터를 더 적게 전송하여 더 많은 정보를 주는 것에 포커스가 되었다. 하지만 이는 근본적으로 해결될 수 없다. 본 제안된 알고리즘은 데이터를 효율적으로 압축함으로서 이를 해결하였다.

1. 서 론

현재 유비쿼터스 센서 네트워크는 무수한 어플리케이션을 목적으로 하여 상용화 하기위해 연구되고 있다. 기존에 산불감지, 식물 생태 관리와 같은 환경 모니터링은 많이 연구되었고 상용화된 예도 있다. 위와 같은 어플리케이션은 USN(유비쿼터스 센서 네트워크)에 극히 일부이고 기본적인 기능이라 할 수 있다. 어떻게 보면 위 적용사례는 근래에 진부적이라고 까지 할 수 있다. 현재에는 BSN(Body Sensor Network) 혹은 BAN[1]이라고도 하는 사용자 몸을 기준으로 하여 하나에 개인망을 이루고 그 망에서 인체에 생체신호 즉 심전도(ECG), 뇌전도(EEG), 맥박, 체온 등을 모니터링하는 일은 센싱되어 획득된 데이터양이 상당히 많다. 이러한 의료데이터는 사람의 생명과 같이 중요한 데이터이기 때문에 데이터를 많이 센싱할 수록 신뢰할 수 있다. 위와 같이 USN이 발전함에 따라서 적용사례가 많아지고 사용자에 요구가 증가됨으로 센싱하여 획득된 데이터량이 점점 많아지고 있다. 이러한 데이터는 더 적게 전송될 수는 없지만 최소에 크기로 조밀하게 전송할 수는 있다. 본 논문은 이를 해결하기 위해 제안한다.

2. 본 론

2.1 제안된 압축기술의 필요성

USN환경에서 사용자의 요구가 많아짐으로써 센싱된 데이터가 많아지고 있다. 이러한 데이터가 많아지면 네트워크 전송데이터가 많아지게 되고 부수적인 문제로 패킷이 손실될 확률 또한 증가하게 된다. 근래에 연구는 이러한 패킷을 덜 전송해 더 많은 정보를 줄 수 없을까하는 것이지만 이는 원천적으로 해결할 수 없다. 본 논문에서는 이러한 문제를 근본적으로 해결할 수 있는 압축기술을 제안한다.

2.2 제안된 압축기술 개요

기존에 압축기술로는 Run-Length 압축 알고리즘, Lempel-Ziv등 대표적으로 있다. 이 알고리즘은 8bit 마이크로 프로세서기반에 수행되기에는 오버헤드가 크다. 또한 더욱 이들 알고리즘을 USN환경에서 사

용하기 힘든 이유는 센서 데이터의 특징적 성격이 다르다는 점이다. 이는 압축효율성을 떨어뜨린다. 센서 데이터 특징적 성격은 아래와 같다.

ⅰ.MCU에 따라서 제한된 데이터크기 즉 Atmelga128L 같은 경우 10bit, MSP430같은 경우 12bit로 한정되어 있다. 데이터 저장은 바이트 단위이기 때문에 기본적으로 Atmelga128인 경우 6bit, MSP430은 4bit가 낭비이다.

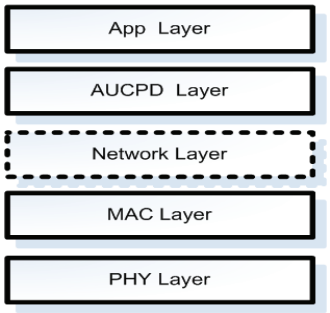
ⅱ.센서에서 생성된 값은 크게 평소값과 이벤트값으로 나눌 수 있다. 예를 들어 화재감지시 화재가 발생하지 않는다면 거의 같은 값의 온도값이 획득되고 화재시에는 평소와 다른 특정 데이터가 획득된다.

위와 같은 USN 환경에 센서 데이터 특징 즉 패턴 때문에 앞에서 언급된 압축기술은 사용될 수 없다.

2.3 AUCPD(Adaptive Ultra-Compact Pattern Data) 알고리즘

제안된 알고리즘은 USN기반에 센싱된 ADC데이터 값을 효율적으로, 최소로 데이터 크기를 줄이기 위해서 고안되었다. 제안된 압축 알고리즘은 앞장에서 언급한 알려진 알고리즘보다 훨씬 우수한 성능을 보이는 것뿐만 아니라 센싱된 ADC데이터 패턴에 따라서 동적인 적용성을 가진다. 이점이 제안된 알고리즘에 강점이라 할 수 있다. 제안된 알고리즘은 버클리 대학의 TinyOS를 기반으로 하여 실제 구현되었다.

2.3.1 AUCPD Network Layer

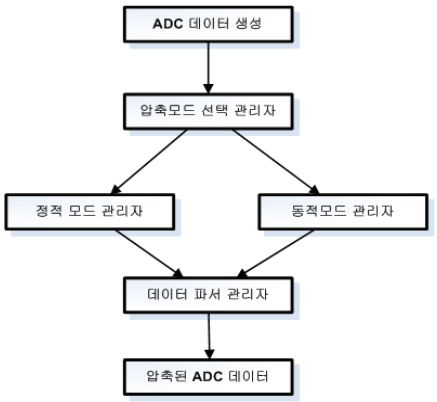


<그림1 TinyOS기반에 AUCPD 계층의 위치>

위 그림은 TinyOS에 네트워크 계층에 제안된 알고리즘을 적용하였다. TinyOS에는 기본적으로 MAC, PHY, APP 계층을 가지며 Network 계층은 TinyOS는 선택사항으로 두고 있다. 제안된 알고리즘은 App와 Network 계층 사이에 혹은 App와 MAC계층사이에 계층을 이룬다.

2.3.2 AUCPD 알고리즘

AUCPD 알고리즘은 크게 네 개의 관리자로 나눌 수 있다. 압축모드 선택 관리자, 정적모드 관리자, 동적모드 관리자, 데이터 파서 관리자로 나눈다. 이렇게 나누는 이유는 구현상 혹은 기능상에 모듈화하기 위함이다. 아래는 위에서 언급한 관리자들 입장에서 본 전체적인 데이터 흐름도이다.



<그림2 AUCPD 알고리즘 전체적인 데이터 흐름도>

ⅰ.ADC에서 센싱된 데이터를 생성

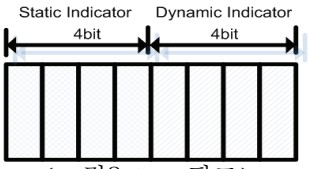
ⅱ. 압축모드 선택 관리자는 생성된 데이터의 패턴을 찾아서 정적 모드 관리자나 동적모드 관리자를 선택, type(Indicator 즉 구분자) 필드 값 생성

ⅲ. 선택된 모드 관리자는 데이터 파서 관리자를 호출하여 관리자 정책에 맞게 데이터 파서 관리자를 호출

ⅳ. 데이터 파서 관리자는 모드관리자에게 호출되어 데이터 구조를 비트로 나누고 모드관리자 명령에 맞게 데이터 쉬프트 해 데이터를 줄임

ⅴ. 압축된 데이터 생성

우선적으로 제안된 알고리즘은 TinyOS 기반으로 적용되었고 TinyOS의 통신 메시지 구조체인 TOS_Msg 구조체를 사용한다. 이 메시지 구조체는 네트워크, ADC, UART등 TinyOS에서 모든 통신을 위한 구조체이다. 제안된 알고리즘은 TOS_Msg 구조체의 선택적으로 사용을 권장하는 type필드값을 사용한다.



<그림3 type필드>

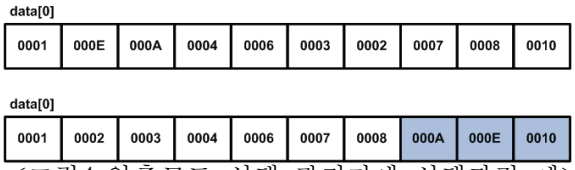
type 필드는 위 그림과 같이 정적모드 관리자가 사용하는 4bit에 Static Indicator와 동적모드 관리자가 사용하는 4bit에 Dynamic Indicator로 나눌 수 있다. 이 필드는 4bit로 0~15까지 표현가능하며 이는 데이터 영역에 데이터 한 개의 크기를 bit로 나타낸다. Static Indicator는 정적모드 관리자에 의해 참조되고 데이터 크기를 명시하며 Dynamic Indicator는 동

적모드 관리자에 의해 참조되고 데이터 크기를 명시한다. 자세한 사용은 다음 내용인 동적 모드관리자에서 설명한다.

다음은 앞에서 언급한 관리자들에 관한 자세한 설명을 한다.

①압축모드 선택 관리자

이 관리자는 ADC에서 생성된 데이터를 검색하여 최대값과 그보다 작은 2개에 값을 찾기 위하여 오름차순 정열을 한다. 8bit 마이크로 프로세서이고 정열은 연산이 많은 과정이므로 연산이 적은 퀵쇼트[2]를 사용한다. 퀵쇼트는 본 알고리즘에 최적이라 생각한다.



<그림4 압축모드 선택 관리자에 선택과정 예>

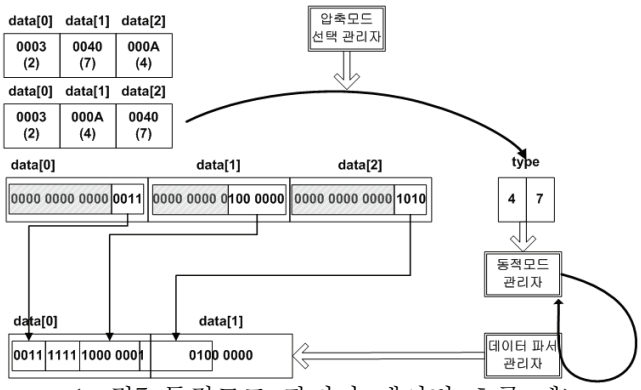
위 그림에서 data는 2바이트 10개에 총 20byte를 예로 들었다. 첫 번째 나오는 data는 정열을 하기전이고 두 번째는 정열 후이다. 위와 같이 정열을 하는 이유는 두 번째에 끝에 3개의 값을 찾기 위함이다. 이 값들에서 자신보다 윈 쪽에 있는 값이 3비트 차이가 나면 동적 모드 관리자를 선택하고 차이가 나이 않으면 정적모드 관리자를 선택한다. 3비트라는 것은 일종 Threshold로써 시스템에 따라서 달라진다. 이렇게 하는 이유는 원본데이터를 비트단위로 표현할 수 있는 최소값을 찾기 위함이고 3개의 최대값을 찾는 것은 너무 큰 값으로 인하여 상대적으로 작은 값들이 큰 값에 데이터 크기 기준이 맞춰지지 않게 하기 위함이다. 위 예를 들어 1, 2, 2, 3, 3, 3, 4, 4, 4, 5 ADC값이 있다하자. 여기서 3, 4, 4, 5 비트크기 차이가 1, 0, 1이 되므로 정적 모드관리자가 선택된다. 선택의 표시로 TOS_Msg의 type 영역에 0x50으로 나타낸다. 즉 Static Indicator에는 5로 데이터가 표현될 수 있는 최대값을 써주고 Dynamic Indicator는 동적 모드관리자는 선택되지 않기 때문에 0을 써준다.

②정적모드 관리자

이 관리자는 압축모드 선택 관리자에게 호출되며 최대값끼리 비트 크기 차이가 3비트이상 나지 않는다. 이 관리자는 data영역을 비트 단위로 인식하고 모든 데이터를 같은 비트 크기로 인식하여 데이터 파서 관리자에게 데이터를 압축시킨다. 예를 들어 0x0001, 0x0007, 0x0005, 0x0010 이라면 가장 큰 데이터인 0x0010의 크기로 비트 크기를 고정시키고 TOS_Msg의 type 필드에는 비트값 0101 0000(0x50)을 표시하고 데이터 파서 관리자를 호출해 데이터를 고정된 비트 크기 5비트로 압축시킨다.

③동적모드 관리자

이 관리자 또한 압축모드 선택 관리자에게 호출되며 호출될 시점은 최대값끼리 비트 크기 차이가 3비트(실험에 의한 threshold) 이상 차이가 날 경우이다. ADC 데이터 특성상 특정 이벤트시 데이터가 극도로 커지는 데 이는 전체 압축 알고리즘에 데이터 크기를 중가 시킨다. 이 관리자는 이를 막기 위함이다. 이는 일종에 센서 데이터 패턴에 맞게 압축을 적용하는 것이다.



<그림5 동적모드 관리자 데이터 흐름 예>

위 그림은 동적모드 관리자에 대한 전체적인 시스템 흐름도이다. 위를 예시로 들어 설명하자면 data영역은 각 2byte 3개의 값이 각각 0x0003, 0x0040, 0x000A가 있다. 압축모드 선택 관리자는 이 데이터를 비트 단위로 환산해서 TOS_Msg에 type 필드를 생성한다. 여기서 데이터값에 비트가 각각 2bit, 7bit, 4bit이다. 만약 정적모드 관리자라면 모두가 7bit에 기준에 맞춰지지만 동적모드 관리자는 2bit과 4bit을 같은 단위로 그래서 type에 4로 나타내고 동적인 값은 가장 큰 값인 7bit으로 써준다. 생성된 type필드만 동적 모드 관리자에게 전달된다. 동적모드 관리자는 type필드를 보고 데이터 파서 관리자를 데이터 하나하나 압축을 진행하면서 주기적으로 데이터 파서 관리자를 호출한다. data영역에 첫 값 0011(0x3) 채우고 담으로 1000 0000(0x40)값을 채우려고 보니 4bit보다 크다 그러므로 type필드에 Dynamic Indicator인 7bit을 기준으로 한다. 하지만 type 필드에 7bit 크기를 사용한다는 일종에 표시를 해주어야 하기 때문에 type 필드에 Static Indicator인 4bit값에 최대값 0xF를 먼저 채워줌으로서 이를 표시한다. 여기서 실체 0xF값이 있을 수 있는 데 이 값은

0xFF로 두 번 표시함으로써 구분한다. 실제 0xF값은 데이터값을 더 늘릴 수 있으나, 그 확률은 1/16이기 때문에 이는 극히 드물고 값이 더 커질 경우는 더 드물게 된다. 이러한 과정을 거침으로서 ADC 데이터 특성상 다양한 크기를 적용성있게 압축할 수 있다.

④데이터 파서 관리자

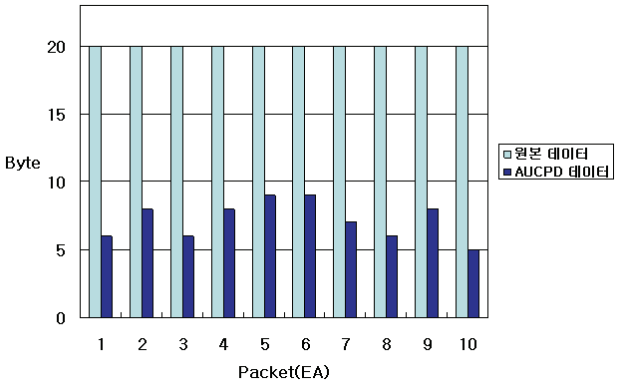
이 관리자는 정적 혹은 동적 모드 관리자에게만 호출되며 데이터영역에 모든 데이터를 비트단위로 환산하여 나누고 삽입하는 과정을 거친다. 실제 데이터를 정책에 맞게 압축하는 것을 담당한다.

2.4 AUCPD 성능평가

본 제안된 알고리즘을 평가하기 위하여 Crowbow사의 micaz을 사용하였다. 이 플랫폼은 8bit MCU인 Atmega128L 기반에 2.4Ghz대역을 갖는 Chipcon사의 CC2420 RF칩을 사용한다. OS는 앞에서 언급한 것과 같이 TinyOS를 사용하였다. 평가를 위하여 간단하게 시나리오를 설정하였다. micaz 노드를 2개를 구성하고 하나는 송신자, 다른 하나는 수신자이면서 호스트와 연결되는 SYN노드를 사용하였다. 송신자는 ECG(심전도)센서를 사용하여 SYN노드에 전송한다. 생성된 데이터량은 4ms당 심전도 센서 ADC값 2byte를 생성한다. 아래와 같은 수식이 적용된다.

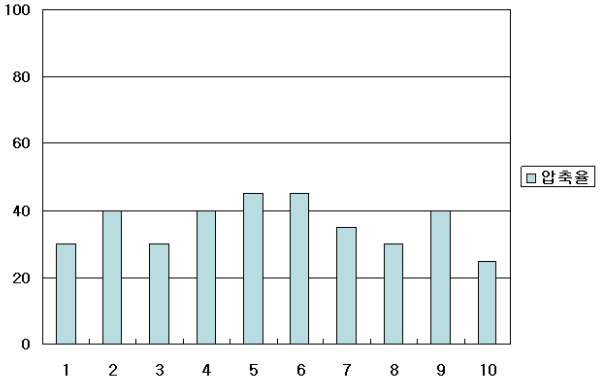
bps(bit per second) = MS X (2 X 8)

MS는 ADC를 데이터를 생성하는 이벤트이고 2 X 8은 2byte를 비트단위로 환산한 것이다. 위 시나리오에 예와 같다면 4ms당 2byte이기 때문에 bps는 4000bps이다.



<그림6 데이터 크기 비교>

위 그래프는 원본 데이터와 데이터 크기를 비교한 것이다. 평균 20byte 데이터를 약 7.2byte의 크기로 줄 일수 있다.



<그림7 AUCPD의 압축율>

위 그래프는 제안된 알고리즘의 압축율을 나타낸 것이다. 평균 약36%의 압축율을 보인다. 이는 제안된 알고리즘이 ADC에서 센싱된 값에 패턴을 때에 따라서 적용성있게 압축한 결과이다.

3 결론 및 개선 사항

3.1 결론

기존에 연구 포커스는 센싱된 데이터를 어떻게 하면 덜 전송해서 더 많은 정보를 줄 수 있을까 하는 것이다. 하지만 본 제안된 알고리즘은 높은 압축율을 바탕으로 위와 같은 방식을 고려하지 않고도 데이터 통신을 원활히 할 수 있다. 이는 희박한 자원을 바탕으로 하는 센서 네트워크에 큰 기여라고 할 수 있다. 본 제안된 알고리즘은 센서 네트워크분야에 기존 방식접근을 새로운 패러다임으로 접근하였다고 할 수 있다.

3.2 개선 사항

본 알고리즘에서는 센싱된 데이터를 기반으로 패턴을 찾아서 그 패턴에 맞게 압축을 하였다. 패턴을 찾는 것이 중요한데 본 알고리즘은 크게 두 가지 동적 모드와 정적 모드로만 나누었다. 하지만 데이터 패턴은 간단하게 두 가지로 나눌 수 없다. 좀 더 많은 패턴을 면밀히 찾아내 효율적으로 압축하는 것이 필요하다. 차후에 더 많은 데이터 패턴을 찾아내 분석 적용할 것이다.

참고문헌

[1] Morchon, O.G., Baldus H., Sanchez, D.S., resource-Efficient Security for Medical Body Sensor Networks, Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop, 4pp, 3-5 April 2006

[2]Quicksort Algorithms ,http://www.softpanorama.org/Algorithms/Sorting/quicksort.shtml