RFID 시스템에서 하이브리드 태그 충돌 방지 알고리즘

정회원 신 재 동*, 여 상 수**, 김 성 권*

Hybrid Tag Anti-Collision Algorithms in RFID System

Jae-dong Shin*, Sang-Soo Yeo**, Jung-Sik Cho* ***Regular Members***

요 약

RFID(Radio Frequency IDentification) 기술은 라디오 주파수를 사용하는 비접촉 자동인식 기술이다. 이런 RFID 기술의 확산을 위해서는 리더(reader)가 다수의 태그(tag)를 짧은 시간 안에 인식하는 다중 태그 식별 문제를 해결 해야만 한다. 지금까지 이 문제를 해결하기 위한 충돌 방지(anti-collision) 알고리즘이 많이 개발되었고 이것들은 크게 알로하(ALOHA) 기반 알고리즘과 트리(tree) 기반 알고리즘으로 나뉜다. 본 논문에서는 이 두 가지 방법의 특징을 혼합한 새로운 충돌 방지 알고리즘 2가지를 제안한다. 그리고 대표적인 충돌 방지 알고리즘인 18000-6 Type A, Type B, Type C, query tree 알고리즘과 성능 비교 및 평가를 한다.

Key Words : RFID, Anti-collision, Hybrid, query tree protocol, ALOHA

ABSTRACT

RFID, Radio Frequency Identification, technology is a contactless automatic identification technology using radio frequency. For this RFID technology to be widely spread, the problem of multiple tag identification, which a reader identifies a multiple number of tags in a very short time, has to be solved. Up to the present, many anti-collision algorithms have been developed in order to solve this problem, and those can be largely divided into ALOHA based algorithm and tree based algorithm. In this paper, two new anti-collision algorithms combining the characteristics of these two categories are presented. And the performances of the two algorithms are compared and evaluated in comparison with those of typical anti-collision algorithms: 18000-6 Type A, Type B, Type C, and query tree algorithm.

Ⅰ. 서 론

RFID(Radio Frequency IDentification) 기술은 RF 신호를 사용하여 물품에 부착된 전자 태그를 식별하는 비접촉 방식의 자동 인식 기술이다[1]. 이러한 RFID 기술의 확산을 위해서는 태그의 저가격, 저전력, 초소형화 문제, 보안 및 프라이버시 문제, 태그 식별자의 코드 표준화 문제 그리고 다중 태그 식별 문제 등의 난제들이 우선적으로 해결되어야 한다. 이 중 다중 태그 식별 문제는 리더의 식별 영역 내에 다수의 태그가 존재할 경우 태그의 정보를 충돌 없이 전송 받아 식별해야하는 리더와 태그 사이의 일 대 다 통신 문제로 정의된다. 이러한 태그 식별 문제는 충돌 방지(anti -collision) 알고리즘을 통하여 해결할 수 있으며 이는 RFID 시스템에서 핵심이 되는 기술이다[2].

충돌 방지 알고리즘은 크게 2가지, 알로하 기반 알고리즘과 트리 기반 알고리즘으로 나누어진다. 알

※ 이 논문은 2005년도 정부(과학기술부)의 재원으로 한국과학재단의 지원을 받아 수행된 연구임(No.R01-2005-000-10568-0).
* 중앙대학교 컴퓨터공학부 알고리즘 및 정보보호 연구실 (mulli2@alg.cse.cau.ac.kr)
** 규슈대학교 정보공학부 (ssyeo@itslab.csece.kyushu-u.ac.jp)
논문번호 : KICS2006-11-496, 접수일자 : 2006년 11월 16일, 최종논문접수일자 : 2007년 3월 9일

로하 기반 알고리즘이란 보통 slotted ALOHA 알고리즘을 말하며 이는 시간을 슬롯(slot) 단위로 나누어 태그를 한 슬롯에 하나만 응답하게 하여 리더가 인식하는 알고리즘이다. 하지만 이 방식은 확률이라는 불확실성에 기초를 두고 있기 때문에 리더 식별 영역 내의 모든 태그를 인식하지 못할 수도 있다.

한편 트리 기반 알고리즘은 태그의 고유한 식별 ID를 사용하여 태그 인식 과정을 진행하면서 트리를 만든다. 이것은 모든 태그를 인식할 수 있고 그 과정을 예측할 수 있는 장점이 있지만 태그들이 많을 때는 트리를 만드는 중 충돌이 많이 발생하여 트리가 깊어지고 결국 태그 인식 시간이 오래 걸리게 된다.

본 논문에서는 대표적인 알로하 기반 알고리즘인 framed slotted ALOHA 알고리즘과 대표적인 트리 기반 알고리즘인 query tree protocol을 결합한 새로운 알고리즘 2가지를 제안한다. 그리고 기존에 RFID에서 사용하는 충돌 방지 알고리즘들과 시뮬레이션을 통해 성능 비교를 한다.

Ⅱ. 관련 연구

2.1 Framed Slotted ALOHA Algorithm

FS-ALOHA 알고리즘[3]은 RFID 시스템에서 태그의 충돌을 해결하기 위해 사용되는 충돌 방지 알고리즘 중 대표적인 알고리즘이다. 이것은 현재 ISO 18000-6 Type A[4]와 Type C[4]가 될 EPCglobal Class 1 Generation 2[5]에서 실제 사용되고 있기도 하다.

FS-ALOHA 알고리즘에서는 리더가 태그에게 ID 전송 요구를 할 때 전체 슬롯 수인 프레임의 크기(frame size; FS)를 함께 전송 한다. 태그는 리더로부터 ID 전송 요구를 수신하면 전송 받은 프레임 크기 안에서 자신이 전송할 슬롯을 랜덤하게 결정하고 자기 차례까지 기다렸다가 전송을 시도한다.

리더에서는 태그에게 ID 전송 요구 후, 3가지 경우가 발생할 수 있다. 먼저 슬롯에 아무런 응답이 없는 경우이다. 이것은 어떠한 태그도 그 슬롯을 선택하지 않는 경우로 무응답(no response)이라 하고 한 프레임 내에서 무응답의 개수를 C_0로 표현한다. 두 번째 경우는 한 슬롯에 한 개의 태그가 응답한 경우이다. 이 경우 전송 중 오류가 없다면 리더는 정상적으로 태그의 ID를 인식할 수 있다. 이후 리더는 이 태그가 다음 프레임에서 대답하는 것을 막기 위해 태그의 상태를 바꾸는 명령을 태그에게 보낸다. 이런 슬롯을 인식(identification)이라 하고 한 프레임 안에서 인식의 개수를 C_1이라고 표현한다.

Forward link	Request	Slot1	Ack 1011	Slot2	Slot3	Slot4	Ack 0111	Request
Return link		1011		Collision		0111		
Tag1				0010				
Tag2						0111		
Tag3		1011						
Tag4				1110				

Frame size = 4

그림 1. FS-ALOHA 알고리즘의 태그 인식 과정의 예

마지막으로 두 개 이상의 태그가 같은 슬롯에 전송을 시도하는 것으로 이 경우 신호 충돌이 발생해 태그들이 전송한 데이터는 잃게 된다. 이것을 충돌(collision)이라 하고 한 프레임 안에서 충돌의 개수를 C_k라고 표현한다.

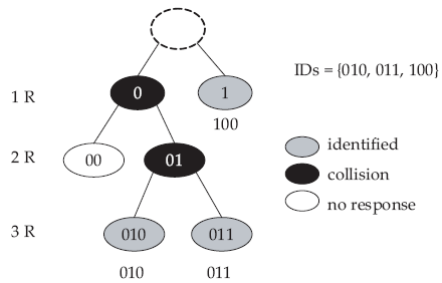
그림 1은 4개의 태그를 이용한 FS-ALOHA 알고리즘의 동작을 나타낸다. 리더는 태그에게 ID 전송 요구와 동시에 프레임의 크기 4를 함께 보내고 태그는 자신이 전송할 슬롯을 선택하여 자신의 ID 전송을 시도한다. 그림 1에서는 *Tag1*과 *Tag4*는 *Slot2*를, *Tag2*는 *Slot4*를, Tag3은 *Slot1*을 선택하였다. *Slot1*과 *Slot4*에서는 하나의 태그만 전송을 시도하였으므로 리더는 *Tag2*와 *Tag3*을 성공적으로 인식하고 태그에게 리더가 인식했다는 것을 알리는 명령 Ack를 보낸다. 이 명령을 받은 태그는 다음 프레임에 참여하지 않는다.

*Slot3*에서는 자신의 ID를 전송한 태그가 없기 때문에 무응답이 되고 *Slot2*에서는 *Tag1*과 *Tag4*가 동시에 자신의 ID를 전송하였기 때문에 충돌이 발생한다. 리더는 한 프레임을 끝내고 인식하지 못한 태그들인 *Tag1*과 *Tag4*에게 ID 재전송 요구를 한다. 이때는 전 프레임에서 나온 무응답 슬롯 수(C_0), 인식 슬롯 수(C_1), 충돌 슬롯 수(C_k)를 사용하여 남은 태그 수를 추정하고 남은 태그 수에 알맞은 프레임 크기로 변경해서 다시 시작한다[6]. 이렇게 진행되어 추정한 태그의 수가 '0'이 되면 FS-ALOHA 알고리즘은 종료된다.

2.2 Query Tree Algorithm

QT 알고리즘[7]은 충돌 방지 알고리즘에서 트리 기반 알고리즘 중 대표적인 것이다. 리더는 태그에게 태그의 ID를 요청할 때 k 비트의 프리픽스(prefix) P_k를 함께 전송한다. 태그는 자신의 ID 앞부분과 프리픽스가 같은 지 확인하고 동일하면 자

신의 ID를 리더에게 응답한다.



Query	0	1	00	01	010	011
Response	collision	identified	no response	collision	identified	identified

그림 2. QT 알고리즘의 인식 과정의 예

이 때 리더의 질의에 대하여 FS-ALOHA 알고리즘과 마찬가지로 어떤 태그도 대답하지 않으면 무응답이, 하나의 태그가 응답하면 인식이, 둘 이상의 태그가 동시에 응답하면 충돌이 발생한다. 리더는 충돌이 일어났을 때 충돌이 발생한 걸 알고, 결국 같은 프리픽스의 태그가 여러 개 있다는 것을 알게 된다. 이 경우 방금 전송한 프리픽스 뒤에 '0'과 '1'을 붙인 2개의 새로운 $k+1$ 비트의 프리픽스 P_{k+1}을 만들어 큐(queue)에 넣는다. 큐에 넣은 프리픽스는 나중에 다시 질의함으로써 태그의 식별 ID를 가지고 트리를 만든다. 큐의 초기 값은 '0'과 '1'이다.

그림 2는 식별 ID가 각각 '010', '011', '100'인 3개의 태그가 있는 상태를 가정하고 QT 알고리즘을 실행한 예제이다. 초기 큐에는 프리픽스 {'0', '1'}이 설정되어 있으며 리더는 큐에서 프리픽스를 꺼내 태그에게 질의한다. 먼저 '0'을 질의 했을 때 태그 '010'과 '011'이 자신의 ID와 프리픽스가 일치하므로 동시에 응답하게 되어 충돌이 발생한다. 이것을 통해 리더는 '0'으로 시작하는 태그가 2개 이상 있다고 보고 큐에 '00'과 '01'을 넣는다. 다음에는 큐에서 프리픽스 '1'을 꺼내와 질의하게 된다. 여기에 맞는 태그는 '100', 1개이므로 이것만 응답하게 되고 리더에게 정상적으로 인식된다. 이렇게 해서 1라운드가 끝나고 전에 큐에 넣은 '00'과 '01'을 대상으로 다시 라운드를 시작하게 된다. 이 알고리즘은 큐가 비었을 때 태그의 ID 트리가 완성된 것으로 보고 종료한다.

Ⅲ. Hybrid Anti-Collision Algorithms

앞서 설명한 FS-ALOHA 알고리즘과 QT 알고리즘은 리더가 인식하려는 태그가 다수인 경우 충돌이 많아 태그 인식 시간을 증가시킨다. 본 논문에서 제안하는 framed query tree (FQT) 알고리즘과 query tree ALOHA (QT-ALOHA) 알고리즘은 FS-ALOHA 알고리즘 과 QT 알고리즘의 혼합으로 태그들을 분산시켜 충돌을 줄임으로서 인식 시간을 줄이는데 목적을 둔다.

3.1 Framed Query Tree Algorithm

FQT 알고리즘은 먼저 태그들을 랜덤하게 프레임 단위로 나눈 후, 이 단위 안에서 각 프레임 별로 QT 알고리즘을 사용해 태그들을 인식 한다. 실제 동작 방식은 다음과 같다. 리더는 태그에게 ID 전송 요구를 할 때 전체 프레임 개수인 에폭 크기(epoch size)를 함께 전송한다. 각각의 태그는 랜덤하게 자신이 참가할 프레임을 결정하고 리더가 자신의 프레임에 대해서 질의할 때만 응답한다. 리더는 각 프레임 안에서 QT 알고리즘을 이용하여 인식 과정을 시행한다. 이 때, 리더는 태그에게 ID의 프리픽스뿐만 아니라 프레임의 번호도 같이 전송한다. 태그는 프레임의 번호가 자신이 선택한 프레임의 번호와 같은 지 확인 한 후 기존의 QT 알고리즘과 동일한 방식으로 전송된 프리픽스를 보고 일치할 때 자신의 ID를 전송한다. 리더는 한 프레임에서 QT 알고리즘 인식 과정을 거쳐 그 프레임에 속한 모든 태그를 인식 한 후에 다음 프레임으로 넘어가게 된다. 이 과정을 모든 프레임에 대해서 반복하여 수행한다.

그림 3의 예제에서는 인식하려는 태그가 8개이고 에폭 크기가 4일 때, FQT 알고리즘의 인식 과정을 보여준다. 먼저 리더는 태그들에게 에폭 크기를 전송해서 랜덤하게 자신의 프레임을 선택한다. 예제에서는 3개의 태그가 *Frame1*을 선택하였고 이들을 대상으로 QT 알고리즘을 사용하여 태그를 인식한다. 그 후 같은 방법으로 다음 프레임으로 진행하면서 태그들을 QT 알고리즘으로 인식 한다.

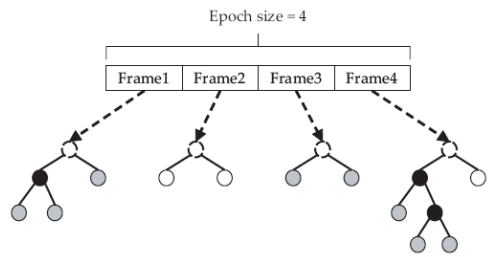


그림 3. FQT 알고리즘 인식 과정 예

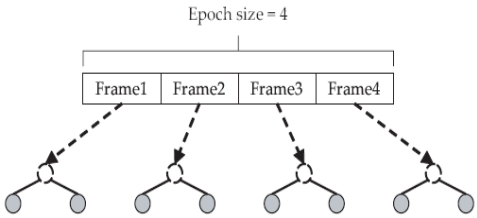


그림 4. FQT 알고리즘 인식 과정의 최상의 예

한편 제안하는 FQT 알고리즘에서는 적절한 에폭 크기를 정하는 것이 성능 향상에 있어서 매우 중요하다. 직관적으로 QT 알고리즘을 시행 했을 때 트리의 깊이가 2 이상이 안 되게 하는 에폭 크기가 가장 좋은 성능을 낸다. 왜냐하면 QT 알고리즘의 초기 큐에 있는 {'0', '1'}을 시행 시 '0'으로 시작하는 태그 하나와 '1'로 시작하는 태그 한 개, 총 두 개의 태그가 인식 되는 것이 이상적이기 때문이다. 그림3에서 *Frame3*이 이런 경우에 해당한다. 인식하려는 태그의 개수가 *N*이고 에폭 크기를 *ES*라고 하면 이상적인 *ES*는 다음 식(1)과 같이 표현할 수 있다.

$$N = 2 \times ES \tag{1}$$

그림 4는 그림 3과 같은 가정에서 이상적인 FQT 알고리즘을 표현한 것이다. 이것은 실제로 4.1절의 시뮬레이션으로도 확인 된다.

그러나 여기에는 큰 문제점이 있다. 얼마만큼의 태그가 있는지, *N*을 모르는 상태에서 태그 인식 과정을 시작하기 때문에 처음부터 알맞은 에폭 크기를 정하는 것이 힘들다. 그러므로 얼마나 많은 태그를 인식해야 하는 지 추정할 수 있어야 하고 추정된 태그 수에 맞게 에폭 크기도 변경할 수 있어야 한다. 그래서 최종적인 FQT 알고리즘은 첫 프레임 테스트(first frame test:FFT)라는 것을 사용한다. FFT는 에폭 크기가 작은데서 시작해서 첫 프레임이 충돌 한계(collision threshold) 이상으로 충돌이 일어나면 인식 과정을 중지하고 에폭 크기를 늘려서 태그들을 재인식하는 것이다. 모든 태그들이 랜덤하게 프레임에 나뉘는 걸 가정하므로 첫 프레임에서 태그의 충돌이 많이 일어난다면 나머지 프레임도 그럴 가능성이 높기 때문이다.

그림 4의 최상의 경우에서 보듯 트리는 태그 2개가 있는 깊이가 1일 때 가장 좋은 성능을 낼 수 있다. 충돌 한계는 태그가 많으면 충돌이 많고 트리의 깊이가 깊어진다는 것을 기반으로 한 한계치이다. 여러 차례 시뮬레이션 한 결과 이 충돌 한계는 3일 때가 FFT 수행 시 알맞은 에폭 크기에 가장 빨리 도달하는 것으로 확인됐다. 3보다 작은 충돌 한계에서는 FFT 수행 중 충돌 횟수에 너무 민감한 나머지 에폭 크기가 적당했어도 이 보다 커져 버릴 수 있고, 반대로 3보다 클 때는 충돌 횟수에 둔감해져 에폭 크기가 느리게 증가 해 알맞은 에폭 크기로 커지는 데 시간이 오래 걸리게 된다. 본 논문에서는 이런 결과를 바탕으로 충돌 한계를 3으로 한다.

3.2 Query Tree ALOHA Algorithm

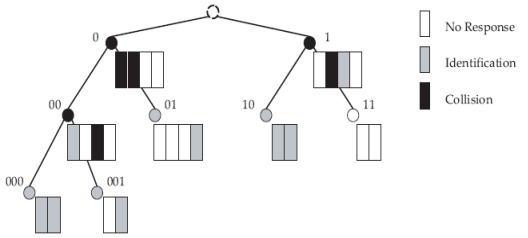
QT-ALOHA 알고리즘은 FS-ALOHA 알고리즘과 QT 알고리즘 혼합의 또 다른 형태이다. FQT 알고리즘의 경우 FS-ALOHA 알고리즘 방식을 기본으로 하고 QT 알고리즘으로 실제 태그인식 과정을 진행한다. 반면 QT-ALOHA 알고리즘의 경우, 이와는 반대로 QT 알고리즘을 기본으로 하며 실제 태그 인식 과정은 FS-ALOHA 알고리즘으로 한다.

리더는 태그에게 ID 전송 요구 시, 프리픽스와 프레임 크기를 같이 전송한다. 이 경우 태그는 자신의 프리픽스와 일치하는 태그들만 전송 받은 프레임 크기를 가지고 FS-ALOHA 알고리즘을 진행한다. FS-ALOHA 알고리즘 진행 중에 한 슬롯에서라도 충돌이 발생하면 QT 알고리즘의 충돌로 인식하여 새로운 프리픽스를 만들어 큐에 넣는다. 이때 기존의 QT 알고리즘과 차이점은 다음 전송할 프레임 크기를 계산하고 이 또한 큐에 넣는다는 것이다. 여기서 프레임 크기 계산[5]은 충돌 난 슬롯의 개수인 C_k를 보고 태그 수를 추정 한다. 이것은 다음 식(2)와 같다.

$$N = 2.3922 \times C_k \tag{2}$$

그리고 추정한 태그 수를 가장 가까운 2의 배수에 맞추어 프레임 크기를 결정하고 이를 큐에 넣는다. 프레임 크기는 2의 배수로 커지기 때문이다. 예를 들어 추정한 태그 수가 30이면 2의 배수 중에 가장 가까운 것은 32로 프레임 크기가 결정된다.

그림 5는 QT-ALOHA 알고리즘의 예제이다. 인식하려는 태그가 8개이고 초기 프레임 크기를 4부터 시작한다고 가정한다. 첫 라운드에 리더는 태그에게 프리픽스 '0'과 프레임 크기 4를 전송한다. 이때 그림 5와 같이 첫 번째와 두 번째 슬롯에서 충돌이 발생할 경우 QT 알고리즘과 같이 '00'과 '01'을 큐에 넣는다.



Query	0	1	00	01	10	11	000	001
Frame Size	4	4	4	4	2	2	2	2
Response	collision	collision	collision	identified	identified	no res.	identified	identified

그림 5. QT-ALOHA 알고리즘의 인식 과정 예

그 후 태그 수를 식 (2)로 추정하고 프레임 크기를 4로 결정, '1'을 큐에서 꺼내 질의한다. 충돌이 없는 노드의 경우 기존 QT 알고리즘과 같이 인식 또는 무응답으로 끝낸다. QT-ALOHA 알고리즘은 이와 같이 진행되며 QT 알고리즘과 같이 큐가 비면 종료한다.

Ⅳ. 시뮬레이션

시뮬레이션에 사용되는 태그 ID는 국제 표준을 따라 64비트 길이로 하고 랜덤하게 생성하였다. 그리고 태그 시뮬레이션 결과의 신뢰도를 높이기 위하여 같은 환경마다 100회의 시뮬레이션을 수행하였다. 본 논문에서 제시되는 결과들은 이 100회의 시뮬레이션 결과에 평균값을 구하고 이를 소수 둘째 자리에서 반올림해서 구한 것들이다.

4.1 Epoch size of FQT Algorithm

첫 번째 시뮬레이션은 태그 100개가 있는 상황에서 에폭 크기를 변경해 가면서 실행했다. 이 결과인 표1을 보면 에폭 크기에 따라 3가지 경우가 나올 수 있다. 첫 번째는 이상적인 에폭 크기인 태그 100개의 절반, 약 50에 가까운 32와 64에서이다. 이 경우 질의-응답 수가 가장 적게 나오고 태그 인식 과정 시간이 짧음을 의미한다. 두 번째로 이상적인 크기 50 보다 작은 경우로 이때는 FFT로 최적의 에폭 크기를 찾아 가지만 FFT를 진행하는 동안 생기는 충돌 때문에 첫 번째 경우보다 성능이 나쁘다. 그리고 마지막으로 이상적인 태그 수보다 많은 경우로 이때는 에폭 크기를 줄이는 과정이 FQT 알고리즘에 없어 초기 에폭을 다 실행해야 하므로 무응답이 많아 질의-응답이 증가한다. 하지만 실제로는 작은 에폭 크기로 시작하므로 문제되지 않는다.

표 1. 에폭 크기(ES)에 따른 성능 비교

Frame Size	Query	C_0	C_1	C_k
1	272.5	49.6	100.0	90.7
16	246.6	55.1	100.0	81.2
32	238.2	56.9	100.0	78.2
64	241.3	85.9	100.0	54.1
128	327.2	193.6	100.0	32.4
256	550.2	430.6	100.0	18.6

4.2 Performance Comparison

두 번째 시뮬레이션은 18000-6 Type A, Type B[4], Type C, 그리고 QT 알고리즘과 제안한 FQT 알고리즘과 QT-ALOHA 알고리즘 간 비교를 했다.

이 비교는 태그 수를 32, 64, 128, 256, 512, 1024로 변경하면서 리더와 태그 사이의 질의-응답 횟수를 비교 수치로 한다. 이것은 알고리즘의 성능을 나타내는 것으로 실제 속도에 가장 큰 요소로 여기서는 속도와 같은 의미로 본다. 시간 수치를 사용하지 않는 이유는 각 표준마다 다르고 또, 각 나라마다도 주파수에 따라 다르기 때문에 정확한 비교 수치가 될 수 없기 때문이다.

한편 Type A, Type C, QT-ALOHA 알고리즘 같은 FS-ALOHA 알고리즘에 기반 한 알고리즘들은 초기 프레임 크기를 태그 수에 상관없이 임의로 32에서 시작하게 했다. FQT 알고리즘에서의 에폭 크기 또한 32에서 시작한다.

그림 6과 그림 7은 시뮬레이션 결과를 나타낸다. 18000-6 Type A와 Type C는 FS-ALOHA 알고리즘으로 동작방법이 같다. 하지만 Type A는 프레임의 크기가 최대 256이 한계라 이보다 많은 태그 수에서 알고리즘을 실행하면 크게 성능이 저하되는 것을 볼 수 있다. 그러나 Type C는 프레임 중간에도 충돌이나 무응답이 많으면 진행하던 프레임을 중단 시키고 새로운 프레임을 진행해서 성능을 향상시킨다. Type B는 트리 계열 중 하나인 binary tree 알고리즘을 사용한다.

결과를 보면 대체적으로 트리 기반 알고리즘들이 ALOHA 기반 알고리즘 보다 질의-응답 횟수가 적게 나타난다. 왜냐하면 ALOHA 기반에서는 한 프레임에서 인식된 태그들이 다음 프레임의 인식 과정에 들어가지 않도록 태그에게 Ack 명령을 보내기 때문이다. 이것은 태그 수만큼 명령을 더 하게 되므로 트리 기반 계열 보다 성능이 좋지 않게 된다.

그림 7은 그림 6의 결과를 태그 수로 나누어 태그 한 개를 인식하기 위해 얼마의 질의-응답이 필요한가를 나타낸 것이다. 이것을 보면 다른 알고리즘

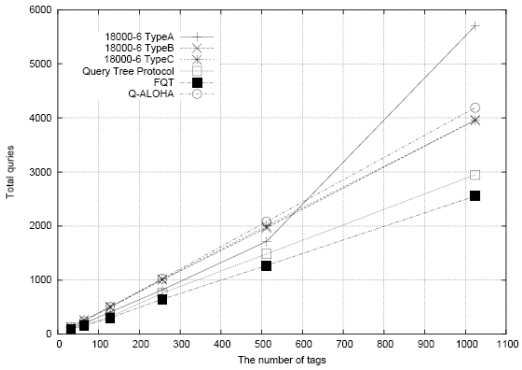


그림 6. 질의-응답 횟수 비교

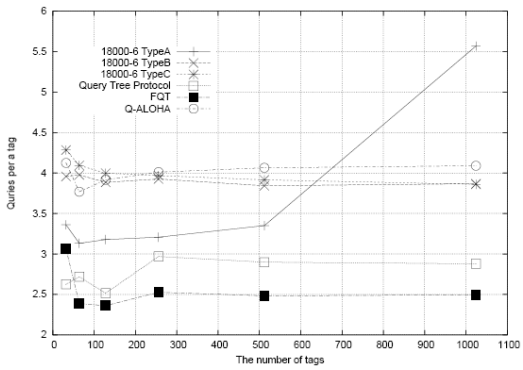


그림 7. 태그 1개 당 인식에 필요한 질의-응답 횟수 비교

들 보다 FQT 알고리즘이 한 개의 태그를 인식하기 위해 가장 적은 질의를 하는 것을 볼 수 있다. 즉, 가장 짧은 시간 안에 리더가 전파 범위 안에 있는 태그들을 인식하는 것이다.

V. 결 론

현재까지는 충돌 방지 알고리즘을 알로하와 트리 계열로 크게 나눠왔지만 본 논문에서는 두 가지 부류의 특징을 결합한 새로운 알고리즘 2가지를 제안하였다. 이 알고리즘들은 FS-ALOHA 알고리즘과 QT 알고리즘을 결합한 형태로 FQT 알고리즘과 QT-ALOHA 알고리즘이다.

이 중에서 특히 FQT 알고리즘은 시뮬레이션 결과 기존에 나온 기타 많은 충돌 방지 알고리즘 보다 질의-응답 횟수에서 10%부터 50%까지의 큰 성능 향상을 보였다. 하지만 FQT 알고리즘은 에폭 크기를 결정하는 첫 슬롯 테스트가 작은데서 커지는 방향으로만 수행할 수 있어 문제가 있다.

향후 연구 방향으로는 FQT 알고리즘에서의 좀 더 빠른 최적의 에폭 크기를 찾는 방법을 연구가 필요하다.

참 고 문 헌

〔1〕 K. Finkenzeller, “RFID handbook”, *John Wiley & Sons*, 1999.

〔2〕 권성호, 홍원기, 이용두, 김희철, “RFID 시스템에서의 트리 기반 메모리래스 충돌방지 알고리즘에 관한 연구”, *정보처리학회논문지* (C), 제 11-C 권 wp6호, December 2004.

〔3〕 F. C. Schoute, “Control of ALOHA Signalling in a Mobile Radio Trunking System”, *International Conference on Radio Spectrum Conservation Techniques*, *IEEE*, pp. 38-42, 1980.

〔4〕 International Organization for Standardization, “ISO/IEC 18000-6:2004/Amd 1:2006”, June 2006.

〔5〕 EPCglobal, “Class-1 Generation-2 UHF RFID Protocol for Communications at 860-960MHz, Version 1.0.9”, 2005.

〔6〕 H. Vogt. “Multiple object identification with passive RFID tags”, *In IEEE International Conference on Systems, Man and Cybernetics (SMC’02)*, October 2002.

〔7〕 Ching Law, Kayi Lee and Kai-Yeung Siu, “Efficient Memoryless Protocol for Tag Identification”, *In Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, ACM*, August 2000.

〔8〕 Jae-Ryong Cha and Jae-Hyun Kim, “Dynamic Framed Slotted ALOHA Algorithms using Fast Tag Estimation Method for RFID System”, *Consumer Communications and Networking Conference, IEEE*, January 2006.

〔9〕 Christian Floerkemeier, and Matthias Wille, “Comparison of Transmission Schemes for Framed ALOHA based RFID Protocols”, *SAINT-W ’06: Proceedings of the International Symposium on Applications on Internet Workshops*, January 2006.

신 재 동 (Jae-dong Shin) 정회원

2005년 8월 : 중앙대학교 컴퓨터공학과 학사 졸업
2005년 9월~현재 : 중앙대학교 컴퓨터공학과 석사과정
<관심분야> RFID 보안, RFID 충돌 방지, 정보보호

여 상 수 (Sang-Soo Yeo) 정회원

1997년 2월 : 중앙대학교 컴퓨터공학과 학사 졸업
1999년 2월 : 중앙대학교 컴퓨터공학과 공학석사
2005년 8월 : 중앙대학교 컴퓨터공학과 공학박사
2006년 3월~2007년 2월 : 단국대학교 강의전임강사
2007년 3월~현재 : 큐슈대학교 정보공학부 방문연구원
<관심분야> RFID 보안, 암호 응용 및 정보보호, 컴퓨터 알고리즘

김 성 권 (Jung-Sik Cho) 정회원

1981년 2월 : 서울대학교 계산통계학과 학사 졸업
1983년 2월 : 한국과학기술원 전산학과 석사 졸업
1990년 8월 : University of Washington 전산학 박사 졸업
1991년 3월~1996년 2월 : 경성대학교 전산통계학과 조교수
1996년 3월~현재 : 중앙대학교 컴퓨터공학과 교수
<관심분야> 생물정보학, 계산기하학, 암호응용 및 정보보호