RFID 충돌 비트를 이용한 다중 태그 인식 알고리즘

지유강* · 조미남* · 홍성수** · 박수봉*

*동신대학교 정보통신공학과, *동강대학 컴퓨터인터넷계열

A New RFID Multi-Tag recognition Algorithm using Collision-Bit

Yoo-Kang Ji* · Mi-Nam Cho* · Sung-Soo Hong** · Soo-Bong Park*

*Dept. of information & Communication Eng. Dongshin University

E-mail : neo@dsu.ac.kr

요 약

RFID(Radio Frequency IDentification)리더는 영역 내에 있는 다수의 태그를 인식할 때 데이터간의 충돌이 발생하는데 이러한 충돌은 리더의 태그인식 시간을 지연시키는 원인이 된다. 리더의 태그인식 시간의 지연을 방지하는 프로토콜은 대표적으로 QT(Query Tree)알고리즘을 사용하는데 본 논문에서는 충돌비트위치를 이용하여 개선된 QT-MTC(Query Tree with Multi-Tag Cognition)알고리즘을 제안한다. 제안 알고리즘은 충돌이 일어난 비트 수와 충돌이 발생한 비트 위치를 스택에 저장하여 트리를 순회하는 횟수를 줄였고, 두 개의 태그를 동시에 식별할 수 있도록 설계되었다. 성능분석 결과 QT 프로토콜에 비교하여 제안알고리즘은 연속된 태그 비트에 대해 높은 성능개선효과를 보였다.

ABSTRACT

RFID(Radio Frequency IDintification) leader is collision of data, when recognizing the multiple tag the inside area. This collision became the cause which delays the tag recognition time of the leader. The protocol which prevents the delay of tag recognition time of the leader the place where representative it uses QT(Query Tree) algorithms, it uses a collision bit position from this paper and are improved QT-MTC(Query Tree with Multi-Tag Cognition) algorithms which it proposes. This algorithm stored the bit position which bit possibility and the collision where the collision happens occurs in the stack and goes round a tree the number of time which, it reduced could be identified two tags simultaneously in order, it was planned. A result of performance analysis, It compared in QT protocols and the this algorithm against the tag bit which is continued a high efficiency improvement effect was visible.

키워드

RFID, 충돌방지, QT알고리즘

Ⅰ. 서 론

RFID(Radio Frequency IDentification)시스템은 무선 환경에서 비접촉 IC카드와 무선IC 태그 등을 활용하여 사물에 부착된 태그에 대한 질의와 응답의 통신을 통하여 데이터처리가 이루어진다. 리더의 안테나에서 전파를 발산하는 동안 태그의 ID와 Data가 저장된 Card(Tag)가 그 마그네틱 필드 내에 접속되면 활성화 되어 자신이 가지고 있는 정보를 안테나로 전송하게 되며 안테나는 태그로 부터 전송된 ID를 Data 신호로 변환하여 컴퓨터에서 약속된 서비스를 제공받을 수 있다. 리더는 태그를 빠르게 인식할 수 있어야 한다.[1] 하나의 리더기에 다수의 태그가 존재하는 환경에서 태그와 태그사이에는 서로 전파를 간섭하는 충돌현상이 발생한다. 데이터 충돌로 인해 리더기는 데이터의 재전송을 요구하게 되고 이러한 데이터 재전송 과정은 대역폭의 낭비와 태그에 부착된 사물의 식별에 소요되는 시간 또한 증가하게 된다. 만약 인식 영역 내에 태그의 개수가 많아질수록 심각해질 수 있다. 따라서 다중 태그 식별의 알고리즘의 구현은 충돌 방지 기법으로

RFID 시스템의 성능 향상을 위해 매우 중요하다.

RFID 시스템에서의 다중 태그 인식 기술은 충돌 현상을 극복하여 다수의 태그를 고속으로 인식하는 기술이다.[2] 다중 태그 인식 기술에서 무기억(Memoless)알고리즘으로 대표적인 알고리즘은 QT(Query Tree)알고리즘[3] 인데 본 논문에서는 태그 간 충돌 여부만을 판단하는 QT알고리즘의 문제점을 극복하여 성능을 개선한 QT-MTC (Query Tree with Multi-Tag Cognition)알고리즘을 제안한다. 제안 알고리즘은 태그 간 충돌이 발생한 비트의 개수와 위치 등의 정보를 추출하고 이를 활용하여 인식 성능을 개선하는 방안으로 설계되었다. QT-MTC는 태그 동작 방식의 변경 없이 리더의 변경만으로 구현한 알고리즘이다.

II. RFID 다중 태그 인식기술

2.1 다중 태그 인식기술의 분류

현재 태그 ID를 전송하는 방식으로 크게 2진 트리 기반의 결정적 알고리즘(tree based algorithm) 방식과 슬롯 알로하 기반의 확률적 알고리즘 (slotted aloha) 방식 등으로 구분할 수 있다. 2진 트리 기반의 결정적 알고리즘은 이진트리를 구성한 후 트리의 노드를 순회하는 방식으로 태그 식별 과정을 예측할 수 있다. 또한, 2진 트리 기반의 결정적 알고리즘은 메모리형 알고리즘과 비메모리형 알고리즘으로 태그 내 메모리 유무에 따라 두 가지 형태로 나뉠 수 있다. 메모리형 알고리즘은 리더기의 질의와 태그응답 등의 상태정보를 저장 관리한다. 비메모리형 알고리즘은 리더기로부터의 질의 값에 의해서만 결정하는 가장 적합한 알고리즘이라 할 수 있다. 슬롯 알로하의 확률적 기반 충돌방지 알고리즘은 슬롯을 임의로 선택하여 식별자를 적재하는 방식으로 한 슬롯에 여러 개의 태그가 전송할 때 충돌 발생을 알려주는데 이는 슬롯 수가 증가하면 프레임 전송시간 또한 증가를 초래할 염려가 있어 태그개수의 정확한 산출이 어렵다. 표1은 지금까지의 충돌방지 알고리즘을 구분하여 대표적인 알고리즘을 정리한 것이다.[2][3]

표 1. 충돌방지알고리즘의 분류

충돌 방지 알고리즘의 분류		
트리 기반의 결정적 알고리즘	메모리형	비트-중재 알고리즘
		분할트리 알고리즘
		bit-by-bit 알고리즘
	비메모리형	트리-워킹 알고리즘
		쿼리트리 알고리즘
		충돌추적트리 알고리즘
슬롯-알로하의 확률적 알고리즘	메모리형	Bit-Slot
	비메모리형	I-Code
		STAC
		Framed Slotted ALOHA
		Dynamic Framed Slotted ALOHA

2.2 QT(Query Tree)알고리즘

Query Tree 알고리즘(Query Tree algorithm)은 트리 기반의 결정적 충돌 방지 방법 중 대표적인 알고리즘이다. QT알고리즘은 태그 응답에 따라 리더가 전송하는 질의가 결정된다. 질의 문자열이 저장된 큐에서 k-비트로 구성된길이의 질의를 가져와 태그들에게 브로드캐스트한다. 각각의 태그들은 리더로부터 전송받은 질의를 자신의 태그 ID와 비트 순서대로 비교한다.[2] 만약 자신의 태그 ID와 수신한 질의 문자열이 일치하면 자신의 태그 ID 전체를 리더기에 전송하고 다음의 질의를 리더로부터 기다린다. 반면 자신의 태그 ID와 수신한 질의 문자열이 일치하지 않으면 매칭 되는 질의 문자열을 수신할 때까지 대기상태로 있는다. 리더에서는 리더기가 보낸 k-비트로 구성된 프리픽스(Bk)와 매칭된 태그들은 자신의 태그 ID의 (k+1)번째 비트부터 마지막 비트까지의 나머지 비트들(Idk+1, I) 리더기로 전송한다. 이때 태그의 응답결과는 크게 세 가지 경우가 존재한다. 첫째, 오직 하나의 태그만 응답하였을 경우 둘째, 태그로부터의 응답이 없을 경우 큐에 저장된 프리픽스를 가져와 새로운 프리픽스(Bk')를 생성한다. 셋째, 다수 개의 태그가 동시에 응답하면서 충돌이 발생할 경우 기존 프리픽스에 '0'과 '1'을 추가하여 각각 큐에 저장한 후 큐에서 새로운 프리픽스(Bk')를 가져와 다음 질의-응답 시 태그에 인자로 전달한다. 이러한 과정은 영역 내의 모든 태그가 식별될 때까지 반복된다. 큐가 비어 있으면 전체 태그 식별 과정은 종료되며 이는 영역 내의 모든 태그가 식별되었음을 의미한다.[2][3]

QT알고리즘에서 초기상태 큐는 null 상태이며, 충돌 발생 시 이전 질의에 '0' 또는 '1'을 추가하여 새로운 질의를 만들기 때문에 충돌이 많이 발생하면 할수록 질의 문자열의 길이는 증가한다.

표 2는 QT 알고리즘의 동작과정 예를 나타낸 것인데 자신의 태그 ID가 000,001,100,101인 4개의 태그에 대해 QT알고리즘을 적용하여 진행 과정을 나타낸 것이다. 맨 윗줄은 리더가 태그에 전송하는 질의 문자열을 나타낸 것이며, 세로줄은 리더가 질의 문자열을 보내고 난 후 충돌 또는 인식된 상태를 나타낸 것이다. 알고리즘에서 큐의 초기화 값은 null 이므로 브로드캐스팅하고 처음 시작에서 모든 태그가 동시에 응답하여 충돌이 발생하였으므로 이전의 질의 문자열에 '0'과 '1'을 추가한 새로운 질의 문자열을 생성하여 큐에 저장한다. 두 번째 단계에서 리더는 큐에서 0을 가져와 태그에게 전송하고 첫 비트가 0인 값에서 다시 충돌이 발생한다. 이와 같은 방법으로 반복 실행하다 질의 문자열이 01이 되었을 때는 모든 태그가 응답하지 않았음으로 큐에서 값을 가져와 다음 질의 문자열로 사용한다.[3][4] 이와 같이 QT 알고리즘을 이용하면 4개의 태그 (000,001,100,101)를 인식하는데 반복 횟수는 11번, 질의 문자열의 총 비트의 수는 59비트이고, 모든 태그들이 전송하는 총 비트의 수는 81비트이다.

표 2. QT알고리즘의 Pseudo Code

[리더]
1. 큐(Q)를 null에 해당하는 ε를 전송하고, 메모리 M에는 null를 전송한다.
2. 질의 문자열 q을 큐(Q)에 pop 한다.
3. 태그 응답에 대하여,
① 하나의 응답을 받은 경우 myID를 메모리 M에 저장한다.
② 하나 이상의 응답을 받았을 경우 큐에 이전의 질의 문자열에 0과 1을 추가하여 저장한다.
③ 태그응답이 없는 경우 2.번부터 반복 실행한다.
4. 큐(Q)가 빌 때까지 2. 번부터 반복 실행한다.

[태그]
1. 자신의 태그 ID를 myID = w1, w2, ... w_{myID_length}
2. 리더기로부터 수신한 질의 문자열을 q라 하고, q의 길이를 q_length 라 한다.
3. 만약 q = ε 또는 q = w1, w2, w3, w4, ... wq_length 이면 전체 태그 ID를 리더로 보낸다.

III. QT-MTC 알고리즘

QT 알고리즘은 하나의 비트에서 충돌이 발생하면 전체 알고리즘에서 충돌이 발생한다고 인식한다. 표 2를 살펴보면 단계 2에서 '0' 질의를 보냈을 때 태그1과 태그2가 동시 응답으로 충돌이 발행함을 알 수 있다. 그리고, 단계4에서 '00' 질의를 보냈을 때 태그1과 태그2가 동시에 응답하여 충돌이 발생하였다.[2] 위와 같이 QT 알고리즘은 한 비트만 충돌하던지 아니면 전체가 충돌이 발생하던지 무조건 충돌로 인식하고 새로운 질의를 생성한다. 본 논문에서 제안하는 QT-MTC 알고리즘은 이러한 문제점을 해결하고 불필요한 리더의 반복 시간을 줄이고 처리 속도를 향상시키고자 한다. QT-MTC 알고리즘에서는 리더가 정확한 비트 위치를 알 수 있다는 점을 이용하여 충돌이 발생한 비트 수와 위치를 큐(Q)가 아닌 스택(Stack)에 저장한다. 또한 질의 문자열을 큐(Q)가 아닌 스택(S)에 저장함으로써 세로방향으로 트리를 순회한다.

표 3. QT-MTC 알고리즘의 Pseudo Code

[리더]
1. 스택(S)을 null에 해당하는 ε를 전송하고, 메모리 M에는 null를 전송한다.
2. 질의 문자열 q을 스택(S)에 Pop 한다.
3. 질의 문자열 q를 브로드캐스트 한다.
4. 태그 응답에 대하여,
① 하나의 응답을 받은 경우 myID를 메모리 M에 저장한다.
② 둘 이상의 응답을 받았을 경우
㉠ 충돌이 일어난 비트열을 R에 전송한다.
㉡ 충돌비트 수를 Cn에 전송한다.
㉢ 충돌이 일어난 비트 위치를 ku에 전송한다.
㉣ 남은 비트의 태그 ID 개수 $2^{(myID_length-k+1)}$를 myIDCount에 전송한다.
㉤ 충돌된 태그 ID 개수 cTagCount를 증가.
㉥ myIDCount와 cTagCount의 값이 같으면 메모리 M에 가능한 태그ID를 전송하고 반복실행을 종료한다.
㉦ Cn이 한 개이면 충돌비트에 0을 설정한 r1과 충돌비트에 1을 설정한 r2등을 설정한 후 R에 전송한다.
㉧ r1과 r2을 메모리 M에 저장한다.
㉨ Cn이 두 개 이상이면 ku-1번째 비트까지의 질의 문자열에 0을 추가한 r1과 1을 추가한 r2를 스택에 Push 한다.
③ 태그 응답이 없는 경우 2.번부터 반복 실행한다.
5. 스택(S)이 빌 때까지 2. 번부터 반복 실행한다.

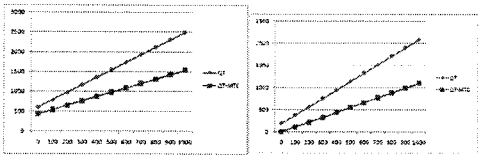
[태그]
1. 자신의 태그 ID를 myID = w1, w2, ... w_{myID_length}
2. 리더기로부터 수신한 질의 문자열을 q라 하고, q의 길이를 q_length 라 한다.
3. 만약 q = ε 또는 q =w1, w2, ... w_{myID_length}이면 전체 태그 ID를 리더로 보낸다.

제안 알고리즘에서 살펴보면 태그 인식 과정은 QT알고리즘에서 알아본 동일한 조건을 적용하였다. 스택(S)의 초기에는 ε가 들어있다고 가정한다. 메모리(M)는 인식된 태그ID를 저장하는 공간으로 이용한다. 리더의 동작은 스택(S)에 저장된 ε를 브로드캐스트 하는 것으로부터 시작된다. 리더는 태그의 응답에 따라 세 가지 경우로 나뉜다. 첫째, 하나의 태그 응답을 전송받은 경우, 해당 태그ID(myID)를 메모리에 저장한다. 둘째, 둘 이상의 응답을 받은 경우에 충돌이 발생하는데 이때 충돌이 발생한 비트 수(Cn)와 처음 충돌이 일어난 비트 위치(ku)를 전송한다. 만일 남은 비트의 태그 가능 ID 개수와 충돌된 태그 ID 개수가 같으면 가능한 태그 ID 메모리에 저장하고 리더의 동작 알고리즘을 끝낸다. 그리고, 충돌이 발생한 비트의 수가 1개이면 0과 1의 값을 충돌이 일어난 비트에 설정한 뒤 메모리에 저장한다. 두 개 이상의 충돌 비트가 감지되더라도 이미 인식된 비트의 위치(ku)-1번째 비트까지의 질의에 0과 1을 추가하여 스택(S)에 저장한다. 마지막으로, 태그 응답이 없는 경우 아무런 동작을 하지 않는다. 리더의 동작 알고리즘은 스택(S)이 빌 때까지 반복 실행한다.

IV. 성능평가

본 논문에서는 실제 물류환경을 고려하여 유사물품의 경우 비슷한 태그 ID를 가지고 있는 것을 확인하였고, 이를 시뮬레이션에 적용시켜 QT 알고리즘과 QT-MTC 알고리즘의 성능을 비교하였다. 그림 1은 리더에서 태그 ID를 인식하는 데 반복 횟수를 QT 알고리즘과 QT-MTC 알고리즘을 적용한 결과를 나타낸 것이다. 자신의 태그 ID를 무작위로 적용시킨 경우와 순차적으로 처리

한 후 알고리즘에 적용시킨 경우를 각각 나타낸 것이다.

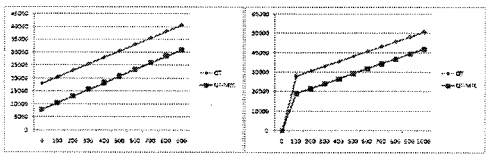


(a) 무작위로 적용 (b) 순차적으로 적용

그림 1. 태그 ID의 적용(반복횟수)

그림 1에 적용 시킨 태그 ID의 길이는 모두 일정하다. QT와 QT-MTC의 알고리즘을 비교한 결과 두개의 차이가 나는 이유는 불필요한 질의 문자열을 전송에 따라 차이가 났음을 알 수 있다. 태그의 개수가 늘어날수록 총 태그를 인식하기 위한 반복 횟수 차이는 QT 알고리즘과 QT-MTC 알고리즘이 서로 벌어진 범위가 넓음을 알 수 있다. (a)보다 (b)에서의 차이가 더 많아짐을 알 수 있는데 이것은 태그 ID를 무작위로 적용시킨 것과 순차적으로 적용시킨 것에서도 영향을 미치는 것을 알 수 있다. 실제 QT-MTC 알고리즘은 QT 알고리즘보다 32.57퍼센트 정도 적은 리더 질의 문자열을 전송하는 것으로 나타났다. 반복 횟수를 줄임으로써 전체 리더가 전송하는 질의 문자열의 양을 줄일 수 있었고 이는 전체 인식 과정의 시간 단축을 기대할 수 있다.

그림 2는 리더에서 태그 ID를 인식하는 데 질의 문자열 비트 수를 QT 알고리즘과 QT-MTC 알고리즘을 적용한 결과를 나타낸 것이다. 자신의 태그 ID를 무작위로 적용시킨 경우와 순차적으로 처리한 후 알고리즘에 적용시킨 경우를 각각 나타낸 것이다.



(a) 무작위 적용 (b)순차적 적용

그림 2. 태그 ID 적용(질의 문자열 비트 수)

전체적인 시뮬레이션 결과는 QT-MTC 알고리즘이 QT 알고리즘에 비해 우수한 성능을 보였음을 알 수 있다. 또한 태그의 개수가 많을수록 QT-MTC 알고리즘의 성능이 더 우수한 것으로 나타났고 이는 불필요한 질의 문자열 수 또한 줄어들고 전체 태그 인식하기 위한 반복 횟수를 감소시킬 수 있었다. 특히 물류 RFID 시스템에서는 비슷한 태그 ID 값을 갖는 다중의 태그들이 동시에 읽혀질 가능성이 크다. 따라서 태그 비트의 중복인 상황을 판단하기 위한 충돌 비트 위치를 저장하는 변수와 충돌 비트 개수를 저장하는 변수, 남은 비트의 태그 가능 ID 개수를 저장할 변수, 충돌된 태그 ID 개수를 저장하는 변수 등이 추가적으로 필요할 뿐이다.

V. 결 론

본 논문은 다중 태그 인식 기술의 대표적인 QT 알고리즘의 성능을 개선한 QT-MTC를 제안하였다. QT-MTC는 충돌 여부만을 판단하는 QT 알고리즘과는 달리 충돌이 발생한 비트의 개수와 위치 등의 정보를 이용하여 트리 순회하는 성능을 개선하는 방안이다. 또한, 반복 횟수를 줄이므로 인해 처리 속도를 올리기 위해 남은 비트의 태그 가능 ID 개수와 충돌된 태그 ID 개수의 정보를 가지고 리더의 질의 횟수를 줄일 수 있다. 시뮬레이션을 통한 검증 결과 QT-MTC는 태그의 수가 많고, 태그 ID 간의 비트 중복성이 높을수록 QT에 비해 우수한 성능을 보였다. 따라서, 물류환경에서 비슷한 상품의 경우 태그들의 ID는 비트가 중복될 확률이 높다는 것을 알고 QT 알고리즘 보다는 QT-MTC 알고리즘에 적용시켜 빠른 태그 식별 인식으로 많은 양의 태그들을 더욱 효율적이고 안정적으로 읽는데 유용하게 사용될 것이다.

참고문헌

[1] 김홍화, "RFID 시스템에서 리더기들간의 협업을 이용한 태그 인식 알고리즘", 淑明女子大學校, December 2006.

[2] 이현지, 김종덕, "충돌 비트 위치를 활용한 RFID 다중 태그 인식 알고리즘", 한국통신학회논문지, Vol. 31, No. 4A, April 2006.

[3] Ching Law, Kayi Lee, and Kai-Yeung Sju, "Efficient Momoryless Protocol for Tag Identifi cation", In Proceedings of the 4th International Workshop on Discrete Algorithms and Metohds for Mobile Computing and Communication, Pages 75-84. ACM. August 2000.

[4] 권성호,홍원기,이용두,김희철, "저비용 RFID시스템에서의 충돌방지 알고리즘에 대한 성능평가", 한국통신학회논문지, Vol. 30, No. 1B, January 2005.