http://dx.doi.org/10.7840/kics.2015.40.10.2035

멀티홉 네트워크에서 생체모방 기반 자원할당 기법

김 영 재◆, 정 지 영*, 최 현 호**, 한 명 훈***, 박 찬 이***, 이 정 륜°

Bio-Inspired Resource Allocation Scheme for Multi-Hop Networks

Young-Jae Kim◆, Ji-Young Jung*, Hyun-Ho Choi**,
Myoung-Hun Han***, Chan-Yi Park***, Jung-Ryun Lee°

요 약

최근 네트워크 단말 수가 증가하고 네트워크 환경이 빠르게 변함에 따라 분산처리 방식의 자원할당 기법이 많이 연구되고 있다. 본 논문에서는 멀티 홉 환경에서 생체모방 알고리즘을 활용하여 분산적인 방법으로 TDMA 자원을 할당받는 Multi-Hop DESYNC 알고리즘(MH DESYNC)을 제안한다. 본 논문에서는 이를 위한 프레임 구조와 자원 할당의 기준 척도가 되는 firing 메시지 구조를 정의하고 관련된 동작 절차를 제안한다. 이를 통해 멀티 홉 환경에서 발생할 수 있는 hidden-node 문제와 firing 신호의 충돌이 발생하였을 때, 충돌 문제를 해결하는 방안을 제시하였다. 모의실험을 통해 멀티 홉 환경에서 제안한 MH DESYNC 알고리즘이 hidden-node 문제를 효과적으로 해결하고 각 노드가 주위 노드와 공평하게 자원을 할당하고 CSMA/CA 알고리즘 보다 데이터 전송율 측면에서 우수한 성능을 나타내는 것을 확인 하였다.

Key Words : Bio-inspired, Resource allocation, TDMA, Distributed, Multi-hop, Ad-hoc Networks

ABSTRACT

Recently, researches on resource allocation algorithms operating in a distributed way are widely conducted because of the increasing number of network nodes and the rapidly changing the network environment. In this paper, we propose Multi-Hop DESYNC(MH DESYNC), that is bio-inspired TDMA-based resource allocation scheme operating in a distributed manner in multi-hop networks. In this paper, we define a frame structure for the proposed MH DESYNC algorithm and firing message structure which is a reference for resource allocation and propose the related operating procedures. We show that MH DSYNC can resolve the hidden-node problem effectively and verify that each node shares resources fairly among its neighboring nodes. Through simulation evaluations, it is shown that MH DESYNC algorithm works well in a multi-hop networks. Furthermore, results show that MH DESYNC algorithm achieves better performance than CSMA/CA algorithm in terms of throughput.

※ 본 연구는 국방과학연구소(ADD-IBR-245)의 지원을 받아 수행되었습니다.
◆ First Author : Chung-Ang University School of Electrical Engineering, youngj89@cau.ac.kr, 학생회원
° Corresponding Author : Chung-Ang University School of Electrical Engineering, jrlee@cau.ac.kr, 종신회원
* Chung-Ang University, jiyoung@cau.ac.kr, 학생회원
** Han-Kyong University, hhchoi@hknu.ac.kr, 정회원
*** Agency for Defense Development, mengddor@add.re.kr, chyipark@add.re.kr, 정회원
논문번호 : KICS2015-09-283, Received September 1, 2015; Revised October 6, 2015; Accepted October 6, 2015

Ⅰ. 서 론

Mesh, sensor, ad hoc 네트워크와 같은 무선 멀티 홉 기반의 네트워크 환경에서는 노드간의 상호 간섭 효과를 고려한 자원의 재사용이 네트워크의 전송 효율을 결정하는 매우 중요한 이슈가 된다. 따라서 무선 멀티 홉 기반 네트워크에서 단말간의 간섭 및 데이터 전송의 충돌을 최소화 하면서 주파수 재사용 계수를 높임으로써 자원 할당의 효율성을 높이기 위한 연구들이 많이 진행되고 있다[1-4]. 데이터 링크의 다중 채널 접근 기법과 연관 지어 자원 할당의 효율성 및 데이터 전송의 충돌 문제를 고찰해 보면, Time Division Multiple Access (TDMA) 와 같은 기법들은 각 노드들이 서로 다른 time slot을 점유하기 때문에 메시지 전송 시 collision free하다는 장점과 traffic load가 많은 환경에서 전체 bandwidth를 충분히 활용할 수 있다는 장점이 있다. 이러한 장점 때문에 무선 멀티 홉 환경에서 TDMA 기반의 무선 자원할당 알고리즘이 연구되어 왔다. 무선 네트워크에서 자원할당 알고리즘들은 크게 중앙 집중적인 방식과 분산적인 방식으로 구분되어 질 수 있다. 중앙 집중적인 방식에서는 기지국과 같은 중앙관리자가 존재하여 전체 네트워크의 자원할당을 관리한다. 이런 중앙 집중적인 방식은 모든 노드의 정보를 알기 때문 자원할당의 효율성을 극대화 할 수 있는 반면, 단말에 수가 증가함에 따라 알고리즘의 복잡도가 올라가고, 네트워크 overhead 및 노드의 power 소비를 증가되는 단점이 있다. 한편 분산적인 방식은 기지국 같은 중앙관리자가 존재하지 않고 각 단말은 서로 자신들의 정보를 교환하고 자원을 할당 받는다. 이러한 분산 처리 기반의 자원 할당 방식은 네트워크 환경이 빠르게 변화는 상황에서 효율적인 자원할당이 가능하다는 장점이 있다. 이에 따라 최근 무선 멀티 홉 기반의 네트워크 환경에서 TDMA기반의 분산적 자원할당 알고리즘에 관한 연구들이 많이 진행되고 있다[3-7].

한편, 최근 생체모방 알고리즘을 활용하여 통신 네트워크의 여러 문제들을 해결하려는 연구가 관심을 모으고 있다.[8-10]. 생체모방 알고리즘은 생태계를 구성하고 있는 각 생물체들의 독자적이면서 단순한 행동규칙을 관찰하여 이를 모델링한 알고리즘으로써, 각 객체들의 행동을 일괄 제어하는 중앙 집중적인 방식과는 달리 각 객체가 간단한 동작 원칙을 독자적으로 수행하는 분산형 알고리즘의 특징을 가진다. 또한 다수의 개체가 간단한 규칙을 통해 동적으로 변하는 외부환경에 안전하고 빠르게 적응하는 특징을 가진다. 이는 통신 네트워크 환경 및 서비스 요구사항과 유사성을 갖는다. 대표적인 예로 생체모방 알고리즘을 동기화 기법에 적용한 반딧불 이론이 있다[11-13]. 반딧불 이론은 초기에 각자 고유의 진동수에 따라 반짝이다가 점차 시간이 지남에 따라 상호작용을 통해 모든 반딧불에 동시에 반짝거리는 현상을 동기화 기법에 적용한 이론이다. 이후 반딧불 이론의 반대 (inverse) 현상을 의미하는 de-synchronization에 관한 연구가 진행되어 왔다. De-synchronization 알고리즘 (DESYNC)은 각 노드가 동시에 반짝거리지 않고 각 노드가 반짝거리는 시간 간격이 되도록 멀리 떨어지도록 설정하여, 결론적으로 모든 노드가 반짝이는 시간 간격이 동일하게끔 수렴하게 된다[14-16]. DESYNC 알고리즘에서 i번째 노드가 반짝이는 (firing) 시간위상(time phase)을 $\phi_i(t)$라 하자. 그러면 자신 바로 전에 반짝인 노드와 와 자신 바로 뒤에 반짝인 노드의 시간위상은 각각 $\phi_{i+1}(t)$와 $\phi_{i-1}(t)$으로 표현되며, 이것의 평균으로 아래 수식(1)과 같이 i번째 노드의 시간위상을 재설정하게 된다. (그림(1) 참조) 이와 같은 동작을 모든 노드가 반복적으로 수행하게 되면 일정 시간이 지난 후에 모든 노드가 동일한 시간간격으로 반짝거리게 된다. 이러한 DESYNC 알고리즘의 특성을 통신 네트워크 분야의 TDMA/FDMA에 적용한다면 네트워크 노드들이 분산적인 방식으로 공평한 자원을 할당 받는 알고리즘을 모델링하기에 적합하다.

$$\phi_i'(t) = (1-\alpha)\phi_i(t) + \alpha\phi_{mid}(t),$$
$$where\, \phi_{mid}(t) = \frac{1}{2}[\phi_{i+1}(t) + \phi_{i-1}(t)] \quad (1)$$

이후 DESYNC 알고리즘에 대한 다양한 후속 연구들이 진행되었다. 2008년도에 R.Nagpal 연구팀은 DESYNC 알고리즘을 멀티 홉 환경에 적용 하려는 연구를 하였다[17]. 연구를 통해 기존 DESYNC 알고리즘을 멀티 홉 환경에 적용하는 경우 hidden-node 문제가 발생할 수 있다는 문제점을 확인 하였다. A.

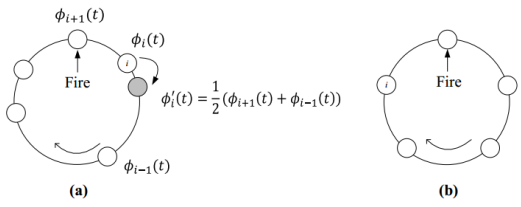

그림 1. DESYNC기법에서 위상 업데이트
Fig. 1. Phase update in DESYNC Algorithm

Motskin 연구팀은 sensor network에서 동기화 및 자원할당을 위한 컨트롤 메시지 (critical message-passing)를 최소화 (extremely lightweight)하기 위하여, 동기화 과정이 필요하지 않고 적은 정보만으로 자원할당이 가능한 Light weighted DESYNC (L-DESYNC) 기법을 제안하였다[18]. 본 연구를 통해 멀티 홉 센서 네트워크 환경에서 두 개의 인접한 노드 간에 할당 받는 자원이 겹치지 않는 것을 확인하였고, 기존의 DESYNC 알고리즘보다 수렴 속도가 빠르다는 것을 확인하였다. 2009년도에 C. Muhlberger 연구팀은 기존의 멀티 홉 DESYNC 기법에서 further work으로 언급한 멀티 홉 환경에서의 hidden-node 문제를 고려한 Extended DESYNC (E-DESYNC) 알고리즘을 제안하였다[19]. 2012년에 C. L. Lien 연구팀은 firing phase를 업데이트 하지 않는 노드 (anchored node)를 두어 DESYNC 되는 수렴속도를 빠르게 하는 Anchored DESYNC (A-DESYNC) 기법을 제안하였다[20]. 이 논문에서는 특정 노드 하나를 anchored node로 정의하여 해당 노드의 firing phase는 특정 값으로 고정되어 있는 상태에서 다른 이웃 노드들의 firing phase만 업데이트하여 DESYNC가 이루어짐을 보였다.

기존의 많은 DESYNC 연구들에서는 fully connected 네트워크 토폴로지를 가정하여 각 노드가 주위 모든 노드의 자원 할당과 관련된 제어 메시지를 동시에 수신할 수 있다고 가정하였다. 또한 멀티 홉 네트워크에서 hidden node 문제를 해결하기 위한 후속 연구들에서는 자신의 2홉 노드의 자원 할당 관련 정보를 얻기 위한 구체적인 알고리즘에 관한 언급이 없고, 실제 알고리즘 구현 시 발생할 수 있는 제어 메시지의 충돌등을 고려하지 않았다. 이에 본 논문에서는 기존 DESYNC 알고리즘을 기반으로 멀티 홉 환경에서 분산적인 방법으로 TDMA 자원할당을 공정하게 하는 Multi-Hop DESYNC (MH DESYNC)알고리즘을 제안한다. 또한 실제 멀티 홉 환경에서 hidden-node 문제를 해결하기 위한 구체적인 프레임 구조 및 제어 메시지 구조등을 제시하고, 메시지 충돌 등을 고려한 제어 메시지 운용 절차를 구체적으로 정의함으로써 실제 적용 가능한 시스템 레벨에서의 연구를 수행한다.

Ⅱ. 제안하는 MH DESYNC 알고리즘

기존 DESYNC 알고리즘에서는 firing 신호를 전송함에 있어, 특정한 control time slot을 사용하지 않고, data time slot을 사용하여 특정한 패턴을 지닌 interrupt 메시지 형태로 전송한다. 본 논문에서는 data time slot을 활용하여 interrupt 메시지 형태로 보내는 firing 신호를 physical firing 신호라고 명명한다. DESYNC 알고리즘에서 physical firing 신호를 송신하는 위치는 매 프레임마다 변경될 수 있기 때문에 주변 수신 노드들이 이를 정확히 예측할 수 없고, 각 노드가 실제 데이터를 전송 중에 삽입되는 제어 메시지 이기에 decoding 지연시간이 생김으로써 실시간 동작을 하기에 한계가 존재한다. 또한 hidden-node 문제를 해결하기 위해서는 자신의 1-hop 주변 노드의 firing 신호 정보를 다시 자신의 주변 노드에게 전송함으로서 모든 노드가 자신의 2-hop 노드의 firing 신호의 위치를 파악해야 한다. 따라서 각 노드는 자신의 firing 신호 외에 주변 노드의 firing 신호를 전송해야 하고, 주변 노드는 이에 따른 firing 신호를 같은 time slot에서 같은 노드의 firing 신호를 중복 수신하거나 혹은 같은 노드의 firing 신호를 다른 time slot에서 여러 개를 수신하게 되는 결과를 초래한다.

이러한 문제를 해결하고자 본 논문에서는 각 프레임을 control time slot과 data time slot을 구분하여 가상 firing 절차를 구현한 새로운 프레임 구조를 제안한다. 제안하는 MH DESYNC 알고리즘에서는 control time slot을 통해 firing 신호를 전송하고 data time slot을 통해서는 실제 data packet을 전송한다. 기존 DESYNC 알고리즘에서는 firing 메시지를 송신하는 실제 time slot의 위치가 자원 할당을 위한 기준이 됨에 비해, 본 논문에서 제안하는 firing 메시지는 control time slot에 송신되며, firing 메시지 안에서 (DESYNC에서처럼) data time slot의 firing 메시지의 위치를 주변 노드들에게 논리적인 형태로 알리게 된다. 이에 우리는 본 논문에서 사용되는 control time slot의 메시지 기반의 firing 신호를 logical firing 신호라고 명명한다. Logical firing 신호 전송을 위하여 각 노드는 매 프레임별 control time slot을 점유하여야 하고, 이를 계속 유지하여야 한다. 이로 인해

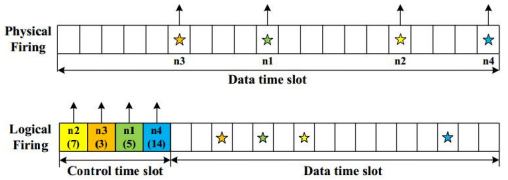

그림 2. Physical firing 신호와 logical firing 신호
Fig. 2. Physical firing signal and logical firing signal

control time slot을 한 번 점유한 노드는 안정적으로 logical firing 신호를 전송할 수 있게 된다. 아래에서 각 프레임 구조와 firing 메시지의 구조를 설명하고 이에 따른 동작 절차에 대해서 살펴보기로 한다.

2.1 프레임 구조

제안하는 프레임 구조는 그림 3에서와 같다. 각 프레임은 C개의 control time slot과 D개의 data time slot으로 구성되어 있다. 각 노드는 control time slot을 1개 점유하고 점유한 control time slot을 매 프레임 사용한다. 노드는 점유한 control time slot을 통해 자신의 firing 신호의 위치 정보와 자신의 1홉 이웃노드들의 firing 신호의 위치 정보를 이웃노드들에게 전송한다. 이웃노드가 control time slot을 통해 전송한 정보를 활용하여 각 노드들은 자신의 2-hop 주변 노드들을 정확히 파악할 수 있다. 이를 통하여 각 노드들은 자신의 firing 시간정보를 업데이트 하고 이에 따라서 data time slot을 점유한다. 각 노드가 2-hop 노드들의 firing 신호 정보를 인지하기 위해서는 2개의 연속된 프레임이 필요하다. 이에 따라 2개의 프레임을 odd와 even 프레임으로 구분하고, 이를 묶어 하나의 슈퍼프레임으로 구성 한다.

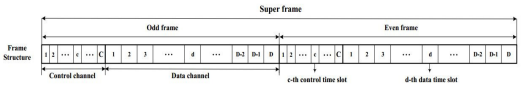

그림 3. 제안하는 프레임 구조
Fig. 3. proposed frame structure

2.2 Firing 메시지 구조

노드는 네트워크 진입시에 control channel에서 하나의 control time slot을 점유한다. 점유된 control time slot을 통하여 각 노드는 firing 메시지를 전송한다. Firing 메시지는 그림 4와 같이 노드의 control time slot 할당 정보를 알리기 위한 C개의 control slot 정보 영역과, 노드의 firing 시간 정보를 알리기 위한 D개의 firing phase 정보 영역으로 구성된다. 각 control slot 정보 영역은 노드 ID와 hop정보 두 가지 부분으로 구성된다. 각 노드는 자신이 점유한 control time slot 정보를 firing 메시지 내의 control slot 정보 영역에 표현한다. 즉 5번 노드가 3번 control time slot을 점유하고 있다면 자신의 ID정보(5번) 및 hop 정보 (0)을 3번째 control slot 정보영역에 표시한다. 또한 자신의 1-hop 노드들이 점유한 control time slot 정보 역시 control slot 정보 영역에 표시한다. 예를 들어 5번 노드의 주변 노드가 1번, 2번 노드가 존재하고, 각각 2번째, 4번째 control time slot을 점유하고 있다면 자신의 control slot 정보 영역의 2번째, 4번째 control slot 정보 영역에 각각 1번, 2번 노드 ID 정보와 hop 정보 (1)을 설정한다. (그림 4 참조) 위의 예에서와 같이 hop정보는 자신이 점유하고 있는 control time-slot에 대한 정보를 기술하면 0으로 이웃노드가 점유한 control time slot에 관한 정보를 표현시에는 1로 설정한다.

각 firing phase 정보 영역도 control slot 정보 영역 같이 노드 ID와 hop정보 두 가지 부분으로 구성된다. 각 노드는 자신이 점유한 firing phase time slot 정보를 firing 메시지 내의 firing phase 정보 영역에 표현한다. 예를 들어 5번 노드가 4번째 firing phase time slot을 점유하고 있다면 자신의 ID정보(5번) 및 hop 정보 (0)을 4번째 firing phase 정보 영역에 표시한다. 또한 자신의 1-hop 노드들이 자신의 1-hop 노드들이 점유한 firing phase time slot 역시 firing phase 정보 영역에 표시한다. 예를 들어 5번 노드의 주변 노드가 1번, 2번 노드가 존재하고, 각각 5번째, 2번째 firing phase time slot을 점유하고 있다면 자신의 firing phase 정보 영역의 5번째, 2번째 firing phase 정보 영역에 각각 1번, 2번 노드 ID 정보와 hop 정보 (1)을 설정한다. (그림 4 참조) 위의 예에서와 같이 hop정보는 자신이 점유하고 있는 firing phase time slot에 대한 정보를 기술하면 0으로 이웃노드가 점유한 firing phase time slot에 관한 정보를 표현시에는 1로 설정한다.

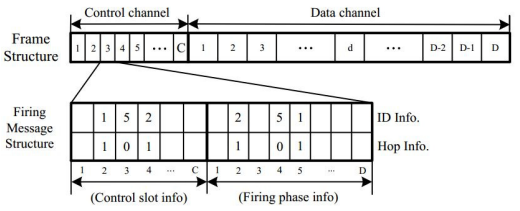

그림 4. Firing 메시지 구조
Fig. 4. Firing message structure

2.3 Logical firing 동작 절차

MH DESYNC에서 각 노드의 logical firing 동작 절차는 그림 5와 같이 i). control time slot 점유, ii) firing phase 점유 및 update, iii). data time slot 점유로 이루어진다. 모든 절차에서 각 노드는 네트워크 초기 진입 시에 하나의 슈퍼프레임을 listen 하여 2hop

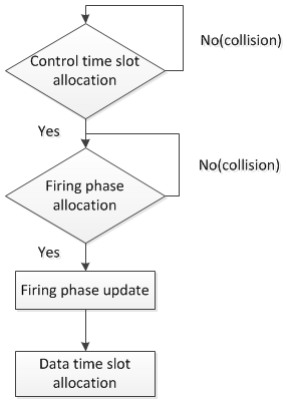

그림 5. Logical firing 동작 절차
Fig. 5. Logical firing operation procedure

이내 이웃노드들의 control time slot 점유 정보 및 firing 신호 위치 정보등을 수신한다. 각 절차에 대해서 자세히 살펴보도록 하자.

2.3.1 Control time slot allocation

(1) 초기 control time slot 할당 과정

노드는 네트워크 초기 진입 시 하나의 슈퍼프레임을 수신한다. 이를 통해 2홉 이내 이웃노드들의 control time slot 점유 현황을 파악하고, 비어있는 control time slot중 하나를 동일한 확률로 정하여 이를 점유한다. 충돌 없이 control time slot을 점유하게 되면, 다음 슈퍼 프레임의 odd 프레임에서 기 점유된 control time slot을 통해 자신의 firing 메시지를 송신한다. 이 후 매 프레임마다 동일한 control time slot을 사용하여 firing 메시지를 송신한다. 만약 control time slot의 점유 과정에서 충돌이 발생했다면 충돌이 발생한 노드는 다음 슈퍼프레임의 odd프레임에서 2hop 이내 이웃노드가 점유하지 않은 control time slot 중 동일한 확률로 하나의 control time slot을 다시 결정한다.

그림 6은 C=4, D=8인 경우 n1, n2, n3 3개의 노드가 control time slot을 점유하려 하는 예를 보여주고 있다. 네트워크 환경은 그림 6(a)와 같고 노드 n2가 이미 4번째 control time slot을 점유하고 있을 때, 노드n1과 n3가 동시에 진입하는 상황을 가정하자. 만약 그림 6(b)와 같이 노드 n1과 n3 서로 다른 control time slot을 점유하려고 한다면 control time slot의 충돌은 발생하지 않고 노드 n1과 n3는 control time slot을 할당 받는다. 하지만 그림 6(c), 그림 6(d)와 같이 노드 n1과 n3가 같은 control time slot을 점유하려고 한다면 노드 n1과 n3의 firing 메시지 충돌이 발생하게 된다.

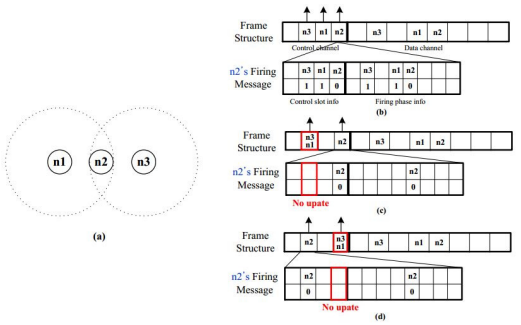

그림 6. control time slot 충돌의 예
Fig. 6. Examples control time slot collisions

(2) Control time slot 충돌 인지 과정

각 노드는 자신의 control time slot 할당의 충돌 유무를 주변 노드의 firing 메시지의 내용을 수신하여 판단하게 된다. 즉, 자신의 control time slot 할당이 성공적으로 이루어진 경우, 자신의 control time slot 할당 정보는 주변 노드의 firing 메시지에 정상적으로 표현된다. (Fig. 6 (b) 참조) 그러나 control time slot의 충돌이 이루어지게 되면 주변 노드는 자신의 control time slot 할당 정보를 할당하지 않게 되어서 이를 통하여 control time slot 충돌을 인지하게 된다. (Fig. 6 (c), (d) 참조)

(3) Control time slot 충돌 인지 유형

Control time slot 충돌 발생 시 해당 firing phase를 결정한 노드가 충돌을 감지하는 경우는 두 가지로 odd프레임에서 충돌을 감지하는 경우와, even프레임에서 충돌을 감지하는 경우로 구분할 수 있다. 그림 6(c)와 같이 n번째 슈퍼프레임의 odd 프레임에서 노드 n1, n3이 동시에 두 번째 control time slot을 통해 firing 메시지를 전송하여 충돌이 발생하면 노드 n2는 자신의 firing 메시지에 두 노드의 정보를 포함하지 않고 n번째 슈퍼프레임의 odd 프레임의 네 번째 control time slot을 통해 전송하게 된다. 따라서 노드 n2의 firing 메시지를 네 번째 control time slot에서 수신한 노드 n1, n3는 자신의 정보가 업데이트되지 않은 것을 통해 자신의 control time slot allocation이 실패하였음을 n번째 슈퍼프레임의 odd프레임에서 바로 인지할 수 있다. 반면, 그림 6(d)와 같이 n번째 슈퍼프레임의 odd 프레임에서 n2는 두 번째 control time slot에서 firing 메시지를 전송하고 n1과 n3가 네 번째

control time slot에서 동시에 firing 메시지를 전송하여 충돌이 발생하는 경우를 고려해 보자. 이 경우 노드 n2는 시간적인 우선함에 의하여 n1, n3 두 노드의 control time slot 점유 정보를 알 수 없기 때문에 자연스럽게 자신의 firing 메시지에 두 노드의 정보를 포함되지 않게 된다. 따라서 노드 n1, n3는 control time slot의 충돌 유무를 n번째 슈퍼프레임의 even 프레임에서 노드 n2의 firing 메시지 안에 자신의 정보가 업데이트되지 않은 것을 통해 인지 할 수 있다. 즉, 충돌이 발생한 두 노드가 점유하려는 control time slot의 위치가 두 노드에 동시에 이웃한 노드가 점유한 control time slot 보다 앞에 있는 경우 n번째 슈퍼프레임의 odd프레임에서 충돌을 감지하고, 뒤에 있는 경우 n번째 슈퍼프레임의 even 프레임에서 충돌을 감지할 수 있다. 따라서 노드들의 control time slot 충돌에서의 일관적인 동작을 위해서, 각 노드는 odd 프레임에서 충돌을 인지하더라도 다음 even프레임에서 firing 메시지를 전송하지 않도록 한다. 이는 충돌이 발생한 두 노드에 동시에 이웃하지 않은 노드는 충돌을 인지하지 못하여 충돌이 발생한 노드의 firing phase 정보를 통해 data time slot allocation을 수행하게 되는데, 이 경우 충돌이 발생한 노드는 실제로 데이터 슬롯을 점유하지 않기 때문에 채널효율이 저하되기 때문이다. 이후 다음 슈퍼프레임의 odd프레임에서 2hop 이내 이웃노드가 점유하지 않은 control time slot 중 동일한 확률로 하나의 control time slot을 결정하여 firing 메시지를 송신한다.

2.3.2 Firing phase allocation and update

(1) 초기 Firing phase 할당 과정

노드가 자신의 control time slot을 점유한 후에는 firing phase 정보 영역을 통하여 다른 노드들에 의해 firing phase로 사용되고 있지 않은 data time slot들을 파악한다. 이 중 하나의 data time slot을 무작위로 선택하고, 선택된 data time slot의 위치를 자신의 가상화된 firing phase (virtual firing phase)로 결정한다.

만약 두 개 이상의 노드가 같은 프레임에서 같은 firing phase 점유를 시도한다면 논리적인 의미에서 firing phase 충돌이 발생하게 된다. 그림7은 firing phase의 점유와 논리적 충돌의 예를 보여주고 있다. 네트워크 환경은 그림 7(a)와 같다. 노드 n2가 먼저 control time slot과 firing phase를 점유한 상태에서 노드 n1과 n3가 진입하여 각각 control time slot을 충돌 없이 할당 받았다고 가정하자. 이런 경우 그림 7(b)와 같이 노드 n1과 n3의 firing phase가 각각 4번째, 2번째로 서로 다른 firing phase를 점유하려고 한다면 논리적인 firing phase 충돌은 발생하지 않고 노드 n1과 n3는 자신이 원하는 firing phase를 할당 받는다. 하지만 그림 7(c), 그림 7(d)와 같이 노드 n1과 n3가 같은 firing phase를 점유 하려고 한다면 노드 n1과 n3은 논리적 의미의 firing phase 충돌이 발생하게 된다.

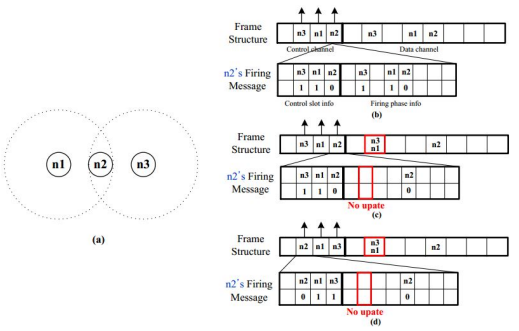

그림 7. firing phase 충돌의 예
Fig. 7. Examples firing phase collisions

(2) Firing phase update 과정

결정된 노드의 firing phase 정보는 다음 슈퍼프레임의 odd 프레임에서 전송된다. 노드 i의 n번째 슈퍼프레임에서의 firing phase를 $\phi_i(n)$이라 하고 $N_2(i)$를 노드 i의 2홉 노드 집합이라 정의하자. 그러면 노드 i의 오른쪽 firing phase reference 노드 $n(i)$와 왼쪽 firing phase reference 노드 $p(i)$는 다음과 같이 정의할 수 있다.

$$n(i) = \arg_{j \in N_2(i)} \min \{ (\phi_j(n) - \phi_i(n)) \bmod D \} \quad (2)$$

$$p(i) = \arg_{j \in N_2(i)} \max \{ (\phi_j(n) - \phi_i(n)) \bmod D \} \quad (3)$$

다시 말해서 노드 n(i) (혹은 p(i))의 n번째 슈퍼프레임에서의 firing phase는 노드 i의 firing phase의 바로 오른쪽 (혹은 바로 왼쪽)에 위치하게 된다. 이 때, 노드 i의 n+1 번째 슈퍼프레임에서의 firing phase는 다음 식에 의하여 업데이트 된다.

$$\phi_i(n+1) = ceil\left(\frac{\phi_{n(i)}(n) + \phi_{p(i)}(n)}{2} \right) \quad (4)$$

다시 말해서 노드 i의 firing phase는 자신의 왼쪽,

오른쪽 firing phase reference 노드들의 firing phase의 중간 값으로 이동하게 된다.

(3) Firing phase 충돌 인지 과정

각 노드는 자신의 논리적 firing phase 할당의 충돌 유무를 주변 노드의 firing 메시지의 내용을 수신하여 판단하게 된다. 즉, 자신의 firing phase 할당이 성공적으로 이루어진 경우, 자신의 firing phase 할당 정보는 주변 노드의 firing 메시지에 정상적으로 표현된다. (Fig. 7 (b) 참조) 그러나 논리적 firing phase의 충돌이 이루어지게 되면 주변 노드는 자신의 firing phase 할당 정보를 할당하지 않고, 이를 통하여 논리적 firing phase 충돌을 인지하게 된다. (Fig. 7 (c), (d) 참조)

(4) Firing phase 충돌 인지 유형

Control time slot 할당 시 충돌 유무의 주변 노드가 인지하는 시점에 차이가 있듯이, 논리적 firing phase의 충돌 역시 같은 설명을 적용할 수 있다. 즉 논리적 firing phase 충돌이 발생한 두 노드가 점유한 control time slot의 위치가 두 노드에 동시에 이웃한 노드가 점유한 control time slot 보다 시간적으로 앞에 위치하고 있는 경우 해당 슈퍼 프레임의 odd프레임에서 바로 충돌을 감지하고, 그렇지 않은 경우에는 해당 슈퍼프레임의 even프레임에서 충돌을 감지할 수 있다. Control time slot 충돌 경우와 마찬가지로 논리적 firing phase의 충돌을 감지한 노드는 해당 슈퍼프레임 even프레임에서 자신의 firing phase 정보를 포함하지 않고 control time slot 정보만 포함하여 firing 메시지를 전송한다. 이는 firing phase 충돌발생으로 인한 data time slot 충돌을 방지하기 위해서이다. 이후 다음 슈퍼프레임의 odd프레임에서 동일한 확률로 하나의 firing phase를 결정하여 firing 메시지를 송신한다.

2.3.3 Data time slot allocation

각 노드는 각 슈퍼프레임별로 자신에게 할당할 자원의 양을 독자적으로 다음과 같은 절차를 통하여 결정한다. 먼저 아래 식을 통하여 자신의 firing phase ($\phi_i(n)$)와 자신의 왼쪽 firing phase reference 노드의 firing phase간의 중간 지점을 구한다.

$$\phi_{i\_l}(n) = ceil\left(\frac{\phi_{p(i)}(n) + \phi_i(n)}{2}\right) \quad (5)$$

또한 자신의 firing phase($\phi_i(n)$)와 자신의 오른쪽 firing phase reference 노드의 firing phase의 중간지점을 아래 식과 같이 구한다.

$$\phi_{i\_r}(n) = ceil\left(\frac{\phi_{n(i)}(n) + \phi_i(n)}{2}\right) \quad (6)$$

노드 i는 n번째 슈퍼프레임에서 $\phi_{i_l}(n)$부터 $\phi_{i_r}(n)$까지 data time slot을 할당 받는다.

Ⅲ. 실 험

제안한 MH DESYNC 기법의 성능 분석을 위하여 먼저 cycle graph 와 unit-disk graph 에서의 동작을 확인하였다. 그림 8에서와 같이 cycle graph는 모든 노드가 자신의 이웃 노드를 2개씩 갖으며 서로 연결되어 있으며, unit-disk graph는 각 노드의 전송반경 내에 있는 노드 간에 대칭적인(symmetric) 직접 통신이 가능하다.

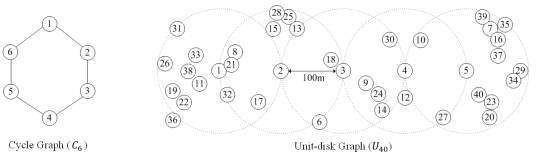

그림 8. 네트워크 배치
Fig. 8. Network topology

3.1 Cycle graph(C_6)에서의 성능 평가

3.1.1 모의실험 parameter

Cycle graph(C_6)의 환경에서 노드 수는 6개, Control time slot의 개수(C)는 5개, data time slot의 개수는 20개로 정의하였다. 모의실험 반복 횟수는 1000번을 수행하며 6개의 노드들이 네트워크에 동시 진입하는 경우 data time slot을 할당 과정을 분석하였다.

3.1.2 모의실험 결과

멀티 홉 환경에서 MH DESYNC 알고리즘을 수행하여 다양한 결과를 얻었다. Cycle graph(C_6)의 환경에서는 크게 그림 9, 그림 10, 그림 11, 그림 12와 같이 4개의 case로 결과를 확인하였다. 그림 9, 그림 10, 그림 11, 그림 12에서 시간이 지날수록 각 노드의 firing phase 수렴하는 것을 확인할 수 있었다. 하지만 초기 값에 따라 data time slot의 할당량이 크게 4가지

형태로 구분된다.

그림 9번에서는 1번째, 3번째 프레임에서 노드 3과 5의 control time slot allocation 시 충돌 발생으로 인해 두 노드의 firing phase allocation 정보가 업데이트 되지 않는 것을 확인 할 수 있다. 그러나 재 할당을 통해 5번째 프레임에서 control time slot/firing phase 할당 절차를 정상적으로 수행한 것을 알 수 있다. 6개의 노드가 자원을 fair하게 할당 받는데 약 10여 프레임이 소요되는 것을 알 수 있다. 그림 9번의 경우 주파수 재사용율이 가장 좋은 경우로, 각 노드는 전체 data time slot 중 1/3씩 할당 받는 것을 확인 할 수 있다. 이런 경우는 서로 3hop 거리에 있는 노드 간에 firing phase가 인접하여 발생하는 경우로, 노드 쌍 (1,4), (2,5), (3,6)이 인접한 firing phase를 점유할 경우 각 노드 쌍은 동일한 왼쪽, 오른쪽 firing phase를 선택하여 자신의 firing phase를 업데이트하여 일정 프레임이 지난 후에 같은 firing phase를 점유하게 된다. 그림 10번에서는 각 노드는 전체 data time slot의 1/4씩 할당 받는 것을 알 수 있다. 이런 경우는 서로 3hop 거리에 있는 노드 간에 firing phase가 인접하여 발생하는 노드 쌍이 2개이고 그렇지 않은 노드의 쌍이 1개인 경우이다. 그림 11번에서는 각 노드는 전체 data time slot의 1/5씩 할당 받는 것을 알 수 있다. 이런 경우는 서로 3hop 거리에 있는 노드 간에 firing phase가 인접하여 발생하는 노드 쌍이 1개인 경우이다. 그림 12번에서는 각 노드는 전체 data time slot의 1/6씩 할당 받는 것을 알 수 있다. 이런 경우는 서로 3hop 거리에 있는 노드 간에 firing phase가 인접하여 발생하는 노드 쌍이 없는 경우이다. 이러한 경우 자원의 재사용은 이루어지지 않았다.

모든 노드가 충돌 없이 control time slot을 할당받는 시간은 최소 3프레임, 평균 6.57프레임이 소요 되

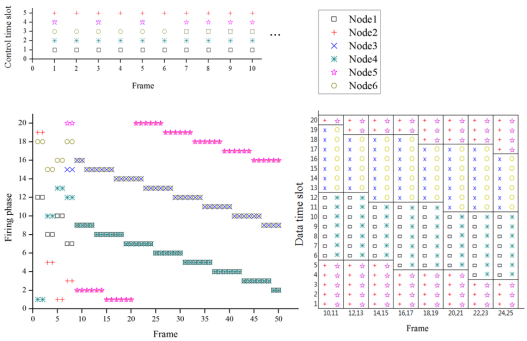

그림 9. 원형 토폴로지에서 모의실험 결과 1
Fig. 9. cycle graph simulation result 1

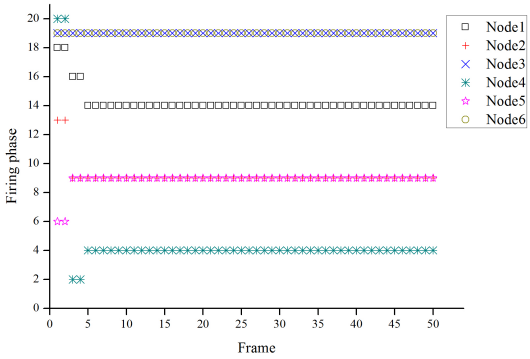

그림 10. 원형 토폴로지에서 모의실험 결과 2
Fig. 10. cycle graph simulation result 2

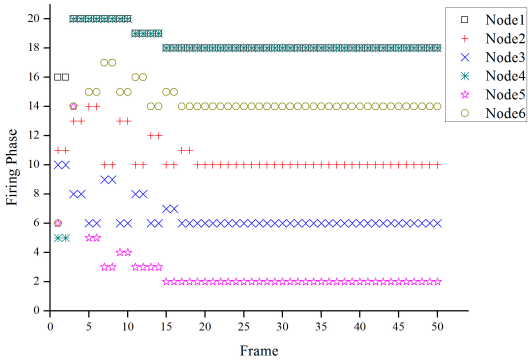

Fig. 11. cycle graph simulation result 3
그림 11. 원형 토폴로지에서 모의실험 결과 3

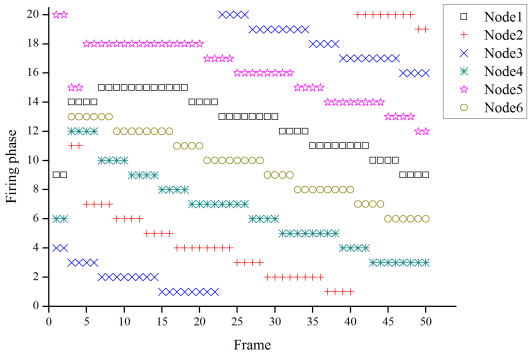

그림 12. 원형 토폴로지에서 모의실험 결과 4
Fig. 12. cycle graph simulation result 4

었고 모든 노드가 충돌 없이 firing phase를 할당 받는 시간은 최소 3프레임 평균 7.28프레임이 소요되었다.

3.2 Unit-disk graph(U_{40})에서의 성능 평가

3.2.1 모의실험 parameter

MH DESYNC의 성능을 unit-dsk graph의 환경에서 모의실험을 통해 성능 분석을 해보았다. 모의실험

parameter는 다음 표 1과 같다. 노드의 진입은 매 슈퍼프레임 마다 한 개의 노드가 순차적으로 진입하는 것을 가정하여 control time slot allocation 충돌은 발생하지 않는다. 또한 무선 멀티 홉 환경에서 성능 평가를 위해 MH DESYNC의 성능을 CSMA/CA 성능과 비교 하였다.

표 1. 모의실험 parameter
Table 1. Simulation parameter

Parameter	Value
The number of nodes	40
The number of control time slots	20
The number of data time slots	40
Transmission rate	1Mbps
Transmission range	100m
Network size	$600\times200m^2$
Packet size	280byte
Packet interval	30ms

3.2.2 모의실험 결과

그림 13은 각 노드의 firing phase가 DESYNC를 이루었을 때, 노드 1과 1의 2hop 이내 이웃 노드의 data time slot 점유현황을 보여준다. 노드1은 15부터 17까지의 data time slot을 점유하였고 자신의 2hop 이내 이웃 노드와 같은 data time slot을 점유하지 않음으로써 hidden-node 문제를 해결하였음을 확인할 수 있다. 노드 1 관점에서 2hop 이내 노드 쌍 (33,3), (26,15), (26,25), (19,25), (19,18), (11,28), (11,13)의 data time slot이 중복되어 할당하였으나, 그림 8에서 알 수 있듯 이 노드 쌍 간의 관계는 모두 서로 3hop 이상 떨어져 있어 주파수 재사용을 하고 있음을 확인할 수 있다.

주파수 재사용양을 확인하고자 노드 i가 할당 받은 data time slot의 수를 B_i로 정의하고 전체 자원을 D로 정의 하였을 때 주파수 재 사용율을 다음과 같이 ($Reusegain = \left(\sum_{i=1}^{N} B_i\right)/D$) 정의 하였다. 그림 14는 그림 8 unit-disk graph (U_{40}) 환경에서 시도 횟수에 따른 reusegain을 나타낸다. 기존의 fully connected 환경에서 각 노드는 전체 자원의 양을 D/N 만큼 할당하기 때문에 reusegain이 1이지만, 멀티 홉 환경에서 각 노드는 2hop 이웃 노드와 자원을 할당받기 때문에 1보다 큰 reusegain을 얻을 수 있다. 시뮬레이션 100번을 수행한 결과 최소 2.525 최대 3.475, 평균 3.057의 reusegain을 얻었다.

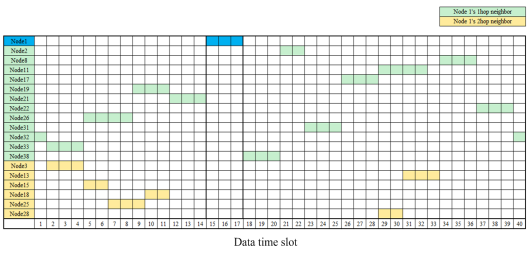

그림 13. Unit-disk graph에서 노드들의 자원할당 예
Fig. 13. Example allocation data time slot in unit-disk graph

제안한 MH DESYNC의 성능을 CSMA/CA 알고리즘과 비교 분석하였다. 성능 평가지표로는 throughput을 사용하였다. Throughput은 단위 시간 동안 destination 노드에 도착한 packet의 평균량으로 정의하였다.

그림 15는 path수에 따른 MH DESYNC기법과 CSMA/CA알고리즘의 평균 수율을 나타내고 있다. 여기에서 source 노드와 destination 노드간의 홉 수는 3홉 이상으로 한정하였고, 라우팅은 AODV를 활용하였다. 그림 8 unit-disk graph (U_{40}) 환경에서 MH DESYNC의 경우 노드 1개당 평균적으로 2.73개의 data time slot을 점유한다. 따라서 path가 1개인 경우 한 프레임에서 전송 할 수 있는 패킷의 평균양은 764.4byte이다. 프레임 길이는 약100ms 이기 때문에 throughput은 평균적으로 61,152bps 나온다. 실제 모의실험 결과에서는 평균적으로 60,788bps정도가 나왔다. MH DESYNC 기법과 CSMA/CA 기법 모두 path의 수가 증가함에 따라 평균 수율이 점차 감소하는 결과를 나타내었다. CSMA/CA기법 보다 제안한 MH

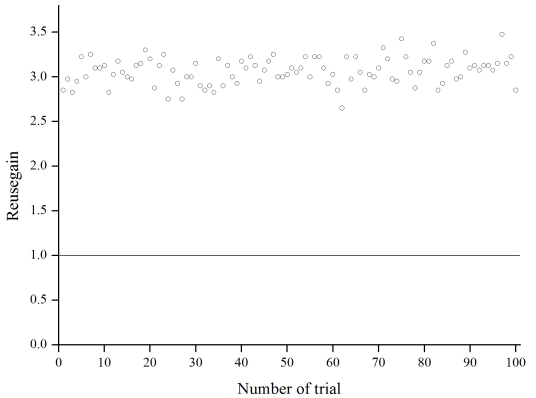

그림 14. 자원의 reusegain
Fig. 14. Reusegain result

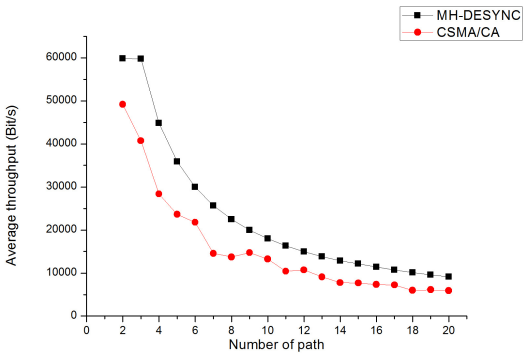

그림 15. MH DESYNC와 CSMA/CA throughput 비교
Fig. 15. MH DESYNC VS CSMA./CA throughput

DESYNC의 평균 수율이 CSMA/CA기법 보다 우수하다는 것을 보여주고 있다. 그 이유는 CSMA/CA 기법은 자원할당을 할 때 노드는 RTS 혹은 CTS 메시지를 보내야 하고, path 수가 증가함에 따라 RTS/CTS 메시지와 back-off 메시지가 증가하기 때문이다. 반면 MH DESYNC의 경우 노드는 매 프레임 자신의 자원을 할당받기 때문에 평균수율은 MH DESYNC기법이 CSMA/CA기법 보다 우수한 성능을 보인다.

Ⅳ. 결 론

본 논문에서는 멀티 홉 환경에서 노드간 분산처리 방식으로 자율적인 자원할당을 할 수 있는 MH DESYNC 알고리즘을 제안하였다. 제안한 MH DESYNC에서는 hidden-node 문제를 해결하기 위한 구체적인 프레임 구조 및 동작 과정을 제시하였다. 제안한 MH DESYNC 방식은 기존의 DESYNC 알고리즘과 달리 실제 data slot에서 물리적 firing 신호를 전송하지 않고, control channel과 firing 메시지를 따로 정의하고, 이를 통하여 2 hop 주변 노드들의 firing phase을 공유할 수 있는 구조를 채택함으로써 데이터 전송의 신뢰성을 높였다. 또한 위와 같은 절차를 적용할 때 control channel 점유 충돌 및 firing phase 충돌의 유형을 살펴보고 이를 극복하기 위한 방안을 제시하였다. 모의실험을 통해 멀티 홉 환경에서 MH DESYNC 알고리즘의 동작을 확인 하였고 firing phase 초기 값에 따라 자원의 재사용양이 달라지는 것을 확인 하였다. 또한 CSMA/CA알고리즘과 MH DESYNC 기법의 성능 비교를 통해 MH DESYNC 알고리즘의 평균수율이 CSMA/CA 알고리즘 보다 우수한 성능이 나타남을 확인하였다. 향후에는 멀티 홉 환경에서 노드의 이동성에 의한 control time slot, firing phase 충돌과 링크 에러에 의한 firing 메시지가 손실 되는 경우를 고려한 MH DESYNC 알고리즘의 성능을 추가적으로 연구할 계획이다.

References

[1] G. Bianchi, L. Fratta, and M. Oliveri, "Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LANs," *PIMRC*, vol. 2, pp. 392-396, Oct. 1996.

[2] J. Lee, J. M. Ahn, K. Lee, and T.-J. Park, "Performance analysis of peer aware communications with CSMA/CA based on overhearing," *J. KICS*, vol. 39B, no. 5, pp 251-259, May 2014.

[3] A. Lozano and D. C. Cox, "Distributed dynamic channel assignment in TDMA mobile communication systems," *IEEE Trans. Veh. Technol.*, vol. 51, no. 6, pp. 1397-1406, Nov. 2002.

[4] Y. Wang and I. Henning, "A deterministic distributed TDMA scheduling algorithm for wireless sensor networks," *WICom 2007*, pp. 2759-2762, Shanghai, China, Sept. 2007.

[5] L. C. Pond and V. O. K. Li, "A distributed time slot assignment protocol for mobile multi-hop broadcast packet radio networks," *MILCOM '89*, vol. 1, pp. 70-74, Boston, USA, Oct. 1989.

[6] C. D. Young, "USAP: a unifying dynamic distributed multichannel TDMA slot assignment protocol," *MILCOM '96*, vol. 1, pp. 235-239, Mclean, USA, Oct. 1996.

[7] C. D. Young, "USAP multiple access: dynamic resource allocation for mobile multihop multichannel wireless networking," *MILCOM '99*, pp. 271-275, Atlantic City, USA, Nov. 1999.

[8] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intell. Mag.*, vol. 1, no. 4, pp. 28-39, Nov. 2006.

[9] M.-J. Kim, J.-H. Choi, and Y.-S. Cho, "Convergence analysis of distributed time and

frequency synchronization algorithm for OFDMA-based wireless mesh networks using bio-inspired technique," *J. KICS*, vol. 39A, no. 8, pp. 88-490, Aug. 2014.

[10] J. Son, S. Shon, and H. Byun, "Bio-inspired energy efficient node scheduling algorithm in wireless sensor networks," *J. KICS*, vol. 38A, no. 6, pp. 528-534, Jun. 2013.

[11] G. Werner-Allen, G. Tewari, A. Patel, R. Nagpal, and M. Welsh, "Firefly-inspired sensor network synchronicity with realistic radio effects," *SenSys '05*, pp. 142-153, New York, USA, Nov. 2005.

[12] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "Desync: self-organizing desynchronization and TDMA on wireless sensor networks," *IPSN '07*, pp. 11-20, New York, USA, Apr. 2007.

[13] J. A. Acebron, L. L. Bonilla, C. J. Perez-Vicente, F. Ritort, and R. Spigler, "The kuramoto model: A simple paradigm for synchronization phenomena," *Rev. Mod. Phys.*, vol. 77, pp. 137-185, 2005.

[14] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "Self-organizing desynchronization and TDMA on wireless sensor networks," *Bio-Inspired Comput. Commun.*, vol. 5151, pp. 192-203, Cambridge, UK, Apr. 2007.

[15] A. Patel, J. Degesys, and R. Nagpal, "Desynchronization: The theory of self-organizing algorithms for round-robin scheduling," *SASO '07*, pp. 87-96, Cambridge, UK, Jul. 2007.

[16] J. Degesys and R. Nagpal, "Towards desynchronization of multi-hop topologies," *SASO '08*, pp. 129-138, Venezia, Italy, Oct. 2008.

[17] A. Motskin, T. Roughgarden, P. Skraba, and L. Guibas, "Lightweight coloring and desynchronization for networks," *IEEE INFOCOM 2009*, pp. 2383-2391, Rio de Janeiro, Brazil, Apr. 2009.

[18] C. Mühlberger and R. Kolla, Extended desynchronization for multi-hop topologies, *Institut für Informatik, Universität Wurzburg, Tech. Rep.* 460, 2009.

[19] C.-M. Lien, S.-H. Chang, C.-S. Chang, and D.-S. Lee, "Anchored desynchronization," *IEEE INFOCOM*, pp. 2966-2970, Orlando, USA, Mar. 2012.

김 영 재 (Young-Jae Kim)

2014년 2월 : 중앙대학교 전자전기공학부 졸업

2014년 3월~현재 : 중앙대학교 전자전기공학과 석사과정

<관심분야> 이동 통신공학, 에드혹 네트워크 등

정 지 영 (Ji-Young Jung)

2011년 2월 : 중앙대학교 전자전기공학부 졸업

2013년 2월 : 중앙대학교 전자전기공학과 석사

2013년 3월~현재 : 중앙대학교 전자전기공학과 박사과정

<관심분야> 이동 통신공학, 에드혹 네트워크, 저 전력 통신 프로토콜 등

최 현 호 (Hyun-Ho Choi)

2001년 2월 : KAIST 전기 및 전자공학과 졸업

2003년 2월 : KAIST 전기 및 전자공학과 석사

2007년 2월 : KAIST 전기 및 전자공학과 박사

2007년 3월~2011년 2월 : 삼성종합기술원 전문연구원

2011년 3월~현재 : 국립한경대학교 전기전자제어공학과 부교수

<관심분야> 차세대 이동통신시스템, 저전력 통신 프로토콜, 매체접속제어, 분산자원관리, 생체모방 알고리즘

한 명 훈 (Myoung-Hun Han)

2007년 2월 : 중앙대학교 컴퓨터공학과 졸업
2009년 8월 : 중앙대학교 컴퓨터공학과 석사
2013년 2월 : 중앙대학교 컴퓨터공학과 박사 수료
2014년 10월~현재 : 국방과학연구소 연구원

<관심분야> 이동통신, 에드혹 네트워크, 생체모방 통신 등

박 찬 이 (Chan-Yi Park)

1998년 2월 : 부산대학교 컴퓨터공학과 졸업
2000년 2월 : POSTECH 컴퓨터공학과 석사
2000년 1월~2002년 1월 : LG 전자기술원 주임연구원
2002년 1월~현재 : 국방과학연구소 선임연구원

<관심 분야> 이동통신, 네트워크 모델링 & 시뮬레이션, 생체모방 통신 등

이 정 륜 (Jung-Ryun Lee)

1995년 2월 : 서울대학교 수학과 졸업
1997년 2월 : 서울대학교 수학과 석사
2006년 8월 : KAIST 전기 및 전자 공학과 박사
2008년 3월~현재 : 중앙대학교 전자전기공학부 부교수

<관심 분야> 저전력 통신 프로토콜, 메쉬 네트워크, 네트워크 이동성, 생체모방 통신 등