한국통신학회논문지 '13-09 Vol.38C No.09
http://dx.doi.org/10.7840/kics.2013.38C.9.764

LTE 펨토셀 네트워크를 위한 적응적 주기의 MLB 알고리즘

김 우 중◆, 이 정 윤*, 서 영 주°

Adaptive Periodic MLB Algorithm for LTE Femtocell Networks

Woojoong Kim◆, Jeong-Yoon Lee*, Young-Joo Suh°

요 약

4세대 셀룰러 네트워크의 데이터양이 증가하면서, 사업자들은 이를 수용하기 위한 네트워크의 용량 문제에 직면하였다. 따라서, 이 문제를 해결하고자 저렴하고 낮은 전력으로 동작하는 펨토셀이 제안되었는데, 이것은 실내 음영지역의 해소 및 사용자의 서비스 품질을 향상시키는 장점을 갖는다. 그러나 펨토셀 네트워크는 소수의 셀에 많은 부하 (Load)가 집중될 가능성이 있다. 이를 해결하고자, 부하 분산 (Load balancing) 알고리즘 중 하나인 MLB (Mobility Load Balancing) 알고리즘이 제안되었다. 이 알고리즘은 부하 분산을 위해 셀 외곽의 사용자를 인접한 셀로 강제 핸드오버한다. 본 논문에서는 주기적으로 동작하는 MLB 알고리즘에서, 주기가 변했을 때 네트워크 성능 지표들이 어떻게 변화하는지를 확인한다. 실험 결과, 짧은 주기로 동작할 때 데이터 차단율이 낮고, 긴 주기로 동작할 때 핸드오버의 빈도가 낮으며 시간당 처리량 (Throughput)이 높은 것을 확인하였다. 이 결과를 바탕으로, 본 논문에서는 적응적 (Adaptive) 주기의 MLB 알고리즘을 제안하였다. 제안된 알고리즘은 긴 주기와 짧은 주기로 동작하는 알고리즘의 장점을 모두 포함하는 것을 실험적으로 확인하였다.

Key Words : LTE, Femtocell, Load Balancing, Self-Optimization Network, MLB

ABSTRACT

The number of users and data packets has increased in 4G cellular networks. Therefore, 4G cellular network providers suffer from the network capacity problem. In order to solve this problem, femtocell concept is suggested. It can reduce the coverage hole and enhance the QoS. However, only small number of femtocells experience the large amount of loads. To solve this problem, Mobility Load Balancing (MLB) algorithm is suggested, which is a kind of load balancing algorithm. To distribute the traffic load, MLB algorithm modifies the handover region. If the handover region is reduced by MLB algorithm, some cell edge users are compulsively handed over to neighbor femtocell. In this paper, we analyze the relation between MLB performing period and performance indicators. For example throughput and blocking probability is reduced, if period is decreased. On the contrast, if period is increased, the number of handover frequency is decreased. Using this relation, we suggest the adaptive periodic MLB algorithm. This algorithm includes the advantage of both long period and short period MLB algorithm, such as high throughput, the small number of handover frequency, and low blocking probability.

※ 본 연구는 미래창조과학부 및 정보통신산업진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(NIPA-2013-H0301-13-3002).
※ 본 연구는 2013년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(2010-0024938).
※ 본 연구는 2013년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 것임(2011-0029034).
◆ 주저자 : 포항공과대학교 컴퓨터공학과, woojoong@postech.ac.kr, 학생회원
° 교신저자 : 포항공과대학교 컴퓨터공학과/정보전자융합공학부 교수, yjseo@postech.ac.kr, 종신회원
* 포항공과대학교 컴퓨터공학과, jylee9@postech.ac.kr
논문번호 : KICS2013-03-151, 접수일자 : 2013년 3월 31일, 최종논문접수일자 : 2013년 8월 12일

Ⅰ. 서 론

최근 LTE (Long Term Evolution)와 같은 4세대 셀룰러 네트워크 (4G cellular network) 사용자는 신규 가입자 및 3세대 셀룰러 네트워크 (3G cellular network) 사용자의 이동으로 인해 그 수가 급격하게 증가하고 있다. 또한, 스마트폰 보급의 확대로 데이터 사용량이 폭발적으로 증가하게 되었으며, 이로 인해 네트워크 사업자는 이를 수용하기 위한 네트워크의 용량 부족 문제에 직면하였다. 이러한 문제를 해결하고자 펨토셀 (Femtocell)의 개념이 제안되었다. 펨토셀은 매크로셀 (Macrocell)보다 좁은 서비스 영역을 제공하며 가정이나 사무실 등 옥내에 설치된 유선 광대역망을 통해 이동통신 코어 네트워크에 접속하는 소규모 기지국이다. 펨토셀은 매크로셀보다 저렴하고 낮은 전력으로 동작하는 장점을 가지며 서비스 영역 및 용량의 확장, 그리고 매크로셀의 부하 (Load)를 분산할 수 있다. 또한, 실내에 설치가 되어 음영지역을 해소하고, 실내 사용자들의 좋은 서비스 품질을 보장한다[1,2]. 이러한 펨토셀의 특징으로 인해, 많은 학자들은 펨토셀의 연구를 활발히 진행하고 있다.

하지만 펨토셀은 소수의 펨토셀에 많은 부하가 집중될 가능성이 있는데, 그 이유는 첫째로 사업자가 아닌, 사용자가 직접 원하는 곳에 펨토셀을 설치할 수 있기 때문이다. 다시 말해 펨토셀은 사업자의 계획 (cell planning) 없이, 사용자가 원하는 곳 어디라도 설치가 될 수 있다. 이것은 펨토셀은 불균일한 분포를 유발할 수 있으며, 이로 인하여 사용자가 균일하게 분포하는 경우에도 소수의 펨토셀에 부하가 집중될 수 있음을 의미한다. 그림 1의 (a)는 이러한 상황을 나타낸다. 그림에서 보는 바와 같이, 사용자들이 균일하게 분포하더라도, ‘Femtocell #3’의 경우에는 부하가 집중되어 4명의 사용자에게 서비스를 해준다. 반면, ‘Femtocell #6’의 경우, 설치된 위치로 인해 단 1명의 사용자에게만 서비스를 제공하게 된다. 이 경우 ‘Femtocell #3’에 많은 부하가 집중된 것을 알 수 있다.

그리고 두 번째 이유는 장소의 특성이다. 장소의 특성은 모든 장소에 동일한 수의 사용자가 존재하지 않는다는 것에 기인한다. 따라서 어떠한 장소는 크게 사람이 많이 집중되는 특성을 갖는 장소와 그렇지 않은 장소, 두 가지로 나눌 수 있다. 사용자가 많이 집중되는 장소는 강의실, 도서관, 또는 회의실 등이 있을 수 있고, 반대로 사용자가 많지 않은 장소는 화장실이나 건물의 복도 등으로 생각할 수 있다. 이러한 장소의 특성 때문에 균일하지 않은 사용자의 분포가 발생 되며, 많은 사용자가 있는 장소에 위치한 펨토셀은 부하가 집중 될 수 있다. 그림 1의 (b)는 그 상황을 설명한 것이다. ‘Femtocell #1’에는 장소의 특성으로 많은 사용자가 존재하는 반면에, ‘Femtocell #3’에는 단 한명의 사용자도 존재하지 않는다. 이 경우 ‘Femtocell #1’에 많은 부하가 집중된 것을 확인할 수 있다.

앞선 두 가지 이유로, 펨토셀은 부하를 분산하는 방법 (Load Balancing)이 적용될 필요가 있다. 하지만 네트워크 사업자들이 모든 펨토셀을 확인하여 부하를 분산할 수는 없는데, 그 이유는 첫째로 펨토셀의 설치 특성 때문이다. 수많은 사용자가 설치한 모든 펨토셀의 존재를 사업자가 인지하여 부하 분

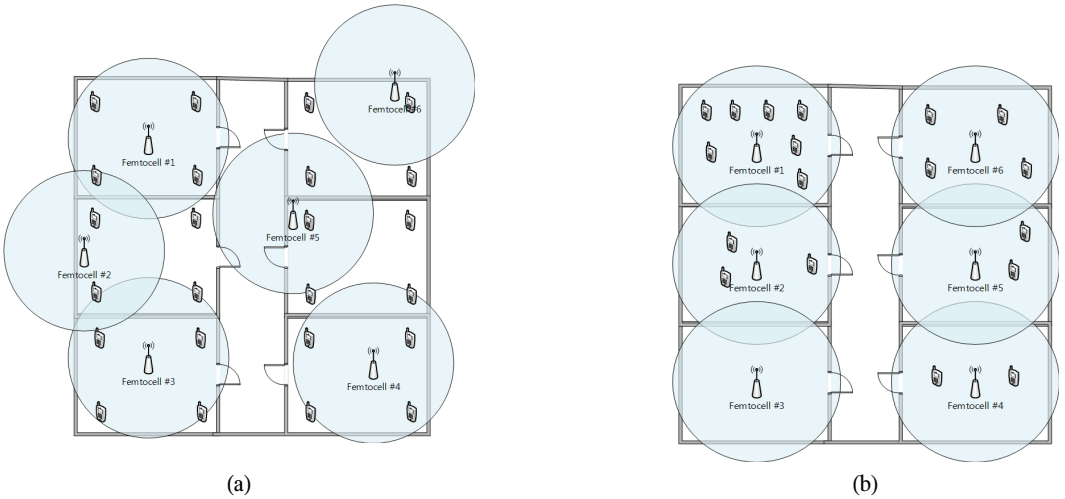


그림 1. 부하 집중의 예시 - (a) 사용자의 배치가 균일한 경우, (b) 펨토셀의 배치가 균일한 경우
Fig. 1. The examples of non-distributed load - (a) Uniform distribution of users, (b) Uniform distribution of femtocells

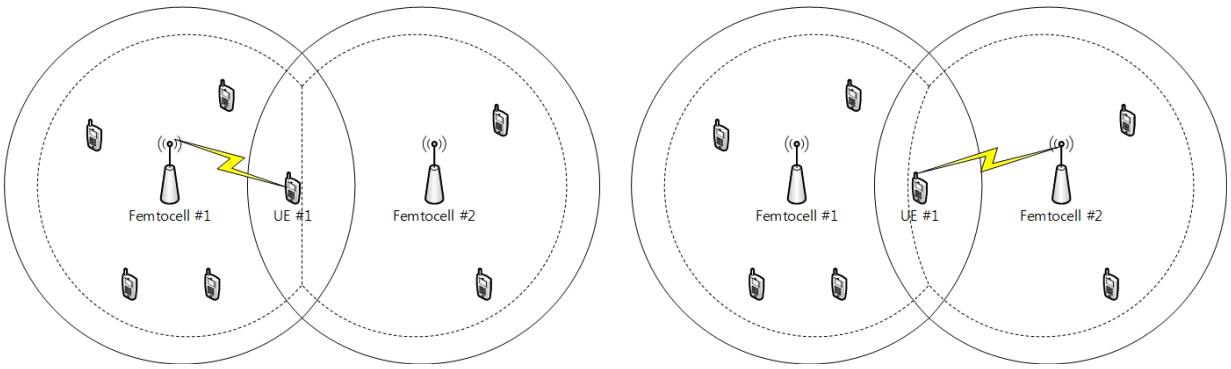


그림 2. MLB 알고리즘의 동작 예시 - (a) MLB 알고리즘 수행 전, (b) MLB 알고리즘 수행 후
Fig. 2. The examples of performing MLB algorithm - (a) Before performing the MLB algorithm, (b) After performing the MLB algorithm

산을 하기는 사실상 어렵기 때문이다. 만일 가능하다 하더라도 필연적으로 운영비용은 증가한다. 두 번째 이유는, 펨토셀이 저마다 다른 환경에서 설치가 되며, 처한 환경이 시간에 따라 변화하기 때문이다. 예를 들어 어떤 펨토셀은 사용자가 오전에 많이 집중되는 장소에 설치된 반면, 또 다른 펨토셀은 점심시간에 집중되는 장소에 설치되었다 하자. 그러면 사업자는 부하 분산을 위해, 매 시간마다 펨토셀의 부하를 확인하여 분산시켜야 하는데, 이는 매우 어렵다. 따라서 펨토셀은 사업자들이 하나하나 조작을 하지 않고, 스스로 환경을 인지하여 부하를 분산할 수 있는 SON (Self-Organizing Network) 방식[3]으로 동작해야 한다.

본 논문에서는, 펨토셀들 간의 부하를 분산하기 위한 SON 방식 기반의 주기적인 MLB (Mobility Load Balancing) 알고리즘에 대해 알아본다. 주기적인 MLB 알고리즘은, 주기적으로 셀에서 처리하는 부하의 양을 인지하여 핸드오버 영역을 조절하는 알고리즘이다. 다시 말해서, 정해진 주기마다 각 셀의 부하를 계산하고, 그 값에 맞게 핸드오버 영역을 증가 또는 감소시키는 방법이다[4,5]. 우리는 이전에 진행한 연구[6]를 확장하여 핸드오버의 빈도, 시간당 처리량, 그리고 데이터 차단율이 주기의 변화와 어떤 관계가 있는지 알아본다. 나아가 그 결과를 바탕으로 새로운 주기적인 MLB 알고리즘을 디자인 할 때의 가이드라인을 제시하며 이를 반영한 적응적 주기의 알고리즘에 대한 성능 평가를 수행한다. 한편, 기존의 연구들[4,5]과는 달리 우리는 펨토셀로만 구성된 펨토셀 네트워크를 대상으로 실험하였다. 그 이유는, 앞서 언급한 바와 같이 펨토셀은 사업자의 계획 없이, 사용자 임의로 설치가 되는 장비이다. 반면 매크로 셀은 통계적인 사용자의 수, 지리적인 특성 등을 고려하여 설치가 된다. 다시 말해서, 매크로셀은 부하 분산을 일부 해소 할, 사업자의 계획에 따라 설치가 되지만, 펨토셀은 그렇지 않다는 것이다. 따라서 우리는 펨토셀 환경에서 기존의 부하 분산 알고리즘을 수행해 보고, 발생하는 양상을 분석하여 펨토셀에 맞는 새로운 알고리즘을 제안한다.

본 논문의 구성은 다음과 같다. II장에서는 논문에서 가정한 환경과, MLB 알고리즘에 대해 설명한다. III장에서는 주기적으로 동작하는 MLB 알고리즘을 펨토셀 환경에서 실험하고, 주기와 네트워크 성능과의 관계를 밝히기 위한 실험 및 결과를 분석한다. IV장에서는 분석된 결과를 반영한 알고리즘을 제안하고 성능을 평가하며, 마지막으로 V장에서 결론을 맺는다.

Ⅱ. Mobility Load Balancing (MLB)

MLB 알고리즘은 셀들의 부하를 분산하기 위해 제안된 방법 중 하나이며, 이 알고리즘은 다음과 같이 동작한다. 먼저 각 셀별로 부하의 양을 계산한다. 계산된 부하의 양이 인접한 셀보다 상대적으로 작다면, 핸드오버 영역을 줄이고, 반대로 계산된 양이 상대적으로 많다면, 핸드오버 영역을 증가시킨다. 이렇게 변화된 핸드오버 영역은 셀 외곽에 위치한 사용자의 강제 핸드오버를 유발한다. 그리고 유발된 강제 핸드오버를 통하여, 해당 셀은 부하가 감소하는 효과를 볼 수 있다. 요약하면 이 방법은 부하가 집중된 셀의 외곽 사용자를 인접 셀로 강제 핸드오버 시켜, 부하의 양을 감소하고자 하는 것이다. 이

알고리즘이 수행되는 예는 그림 2와 같다.

그림 설명에 앞서, 실선은 실제 전송 가능한 범위를 의미이며, 점선은 MLB 알고리즘에서 조정하는 핸드오버 영역을 의미한다. 그리고 부하는 한 펨토셀에서 서비스 하는 사용자의 수라고 정의하고, 알고리즘 수행 전, 'UE #1'은 'Femtocell #1'에서 서비스를 받고 있다고 하자. 이 상황에서 MLB 알고리즘이 수행되면, 먼저 각 셀의 부하가 계산된다. 그 결과, 'Femtocell #1'의 부하는 'Femtocell #2'보다 높음을 알 수 있다. 따라서 'Femtocell #1'은 자신의 부하를 줄이기 위해, 핸드오버 영역을 감소시키고, 반면에 'Femtocell #2'는 자신의 부하가 상대적으로 적기 때문에 핸드오버 영역을 증가시킨다. 그러면 그림 2의 (b)의 그림과 같이, 사용자 'UE #1'은 'Femtocell #1'에서 'Femtocell #2'로 강제 핸드오버가 되며, 그 결과 부하의 양이 많았던 'Femtocell #1'은 부하가 감소하였다.

이러한 MLB 알고리즘 종류 중 하나로, 정해진 수행 주기를 갖는 주기적인 MLB 알고리즘이 제안되었다[5]. 이 알고리즘은 정해진 주기마다 부하의 양을 계산하며 그 결과를 바탕으로 핸드오버 영역을 조정하는 방법이다. 알고리즘의 내용은 다음과 같다.

우선 핸드오버는 식 (1)의 조건이 만족될 때 수행이 된다.

$$M_j - M_i > O_i^{(cs)} + O_{i,j}^{(cn)} + \xi + \eta \tag{1}$$

식 (1)에서, M은 각 셀의 RSRP (Reference Signal Received Power)값이며, ξ와 η은 각각 핸드오버의 마진값 (Hysteresis Margin)과 고정된 상수를 각각 의미한다. 그리고 i는 현재 사용자가 서비스를 받고 있는 셀의 인덱스이며, j는 인접한 셀의 인덱스이다. $O_i^{(cs)}$는 서비스를 제공하는 셀의 특징을 반영하여 조절 가능한 핸드오버 상수 값이다. 다시 말해서 서비스를 제공하는 셀의 정보만을 바탕으로 핸드오버를 결정하는 상수 값이다. 반면 $O_{i,j}^{(cn)}$ 값은 서비스를 제공하는 셀과 인접한 셀들 사이의 상호작용을 기반으로 정의가 되는 상수 값이며, [5]에서는 (2)와 같이 정의하였다. 이 식은 알고리즘 동작 중에 발생한 핸드오버가 받은 신호의 세기뿐만 아니라 각 셀들의 부하도 고려한 것을 알 수 있다.

$$O_{i,j}^{(cn)} = \begin{cases} min(O_{i,j}^{(cn)} + \Delta, O_{max}^{(cn)}), & if \rho_j - \rho_i \geq \rho_{th} \\ max(O_{i,j}^{(cn)} - \Delta, -O_{max}^{(cn)}), & if \rho_i - \rho_j \geq \rho_{th} \\ O_{i,j}^{(cn)}, & if\ |\rho_i - \rho_j| < \rho_{th} \end{cases} \tag{2}$$

식 (2)에서, Δ는 한번 알고리즘을 수행했을 때 $O_{i,j}^{(cn)}$가 변화하는 오프셋 (Offset)이고, $O_{max}^{(cn)}$은 $O_{i,j}^{(cn)}$이 최대로 가질 수 있는 한계치 (Threshold)를 의미한다. 그리고 ρ는 부하를 나타내는 변수이다. ρ_i의 경우, 현재 서비스를 받는 셀의 부하를 나타내고, ρ_j는 인접한 셀 j의 부하를 나타낸다. 마지막으로 ρ_{th}는 부하의 한계치를 의미한다. 이 식은, 두 셀 사이의 부하 차이가 ρ_{th}보다 큰 경우에, 높은 부하를 갖은 셀의 핸드오버 영역을 Δ만큼 감소시킨다는 의미이다. 반대로 낮은 부하를 갖는 셀은 Δ만큼 핸드오버 영역을 증가시킨다는 것이다. 그리고 여기에서 사용된 ρ는 아래의 식 (3)으로 정의한다.

$$\rho_i = \sum_k (K_c, \frac{n_k}{K\Delta T} \frac{R_k^{(req)}}{T_k}) \tag{3}$$

식 (3)에서, k는 셀 i에 연결된 베어러 (Bearer)의 번호를 의미한다. 그리고 K는 하나의 서브 프레임 (Subframe) 안에 있는 PRB (Physical Resource Block)의 수를 의미한다. 또한 ΔT는 한 주기동안 사용 된 서브 프레임의 수를 의미하며, n_k은 실제 베어러 k가 사용한 PRB의 수를 의미한다. T_k는 한 주기동안 측정 된 실제 시간당 처리량 (Throughput)이고, $R_k^{(req)}$는 최소한으로 보장해야 할 전송율 (GBR: Guaranteed Bit Rate)이다. K_c는 한 베어러가 갖는 부하의 한계를 의미한다.

주기적인 MLB 알고리즘의 동작은, 주기를 설정하고, $O_{i,j}^{(cn)}$값을 모두 0으로 초기화 하는 것으로 시작한다. 그리고 정해진 주기가 되면, 먼저 주기마다 부하의 양인 ρ값을, 식 (3)을 사용하여 각 셀별로 계산한다. 이 후 계산된 결과를 이용하여, 식 (2)를 통해 $O_{i,j}^{(cn)}$를 갱신한다. $O_{i,j}^{(cn)}$값의 변화는, 식 (1)에 따라 핸드오버 조건의 변화를 유발하며, 사용자들은 변화된 조건에 맞게 다시 핸드오버를 수행하게 된다. 그리고 점진적인 $O_{i,j}^{(cn)}$값의 증감으로 인해, 셀 외곽의 사용자부터 강제 핸드오버가 된다. 이러한 내용은 알고리즘 1처럼 서술할 수 있다.

한편, 이 알고리즘은 중앙 집중식 알고리즘 (centralized algorithm)이다. 다시 말해, 미리 선정된

알고리즘 1. 주기적인 MLB 알고리즘[6]
Algorithm 1. MLB algorithm with static period[6]

```
Initialize all O_{i,j}^{(cn)} values to 0;
Set period of algorithm;
while (every periods) {
    Calculate the ρ_i for each cell;
    Update O_{i,j}^{(cn)} value for each cell;
}
```

한 개의 펨토셀이 부하 분산을 결정하는데 필요한 정보인 식 (2)의 결과를 주기마다 전달 받고, 이 정보를 바탕으로 식 (3)을 이용해 핸드 오버 영역을 결정하게 된다. 이 후, 해당 결과를 다시 각 셀들에게 전달해 준다. 이 때, 정보 교환은 인터넷 망이나, LTE에서 정의한 X2 인터페이스를 통해 교환한다.

Ⅲ. 주기와 알고리즘의 관계

3.1. 실험 환경

본 논문은 펨토셀 네트워크를 대상으로 실험을 진행하였다. 펨토셀 네트워크란 특정 공간 내에 펨토셀들만이 존재하는 환경이다. 다시 말해서 어떤 매크로셀의 영향을 받지 않고 펨토셀로만 구성된 환경을 의미한다. 기존 연구들[4,5]과 달리 펨토셀 네트워크로 실험한 이유는, 앞서 I장에서 언급 한 부하 집중 현상이 매크로셀보다 자주 그리고 심화되어 발생하기 때문이다. 왜냐하면, 매크로셀의 경우 지리적 특성 및 통계적인 사용자 수가 고려된 사업자의 계획 (cell planning)을 바탕으로 설치가 되는 반면, 펨토셀은 사업자의 계획 없이 사용자 임의로 설치가 된다. 즉 어떠한 부하 분산의 고려도 없이 설치되는 펨토셀에서 부하 집중현상이 매크로셀보다 심화되어 발생할 것으로 예상되기 때문에 우리는

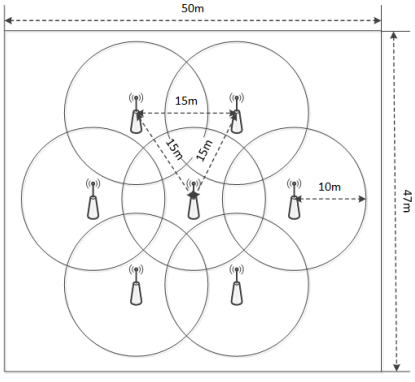


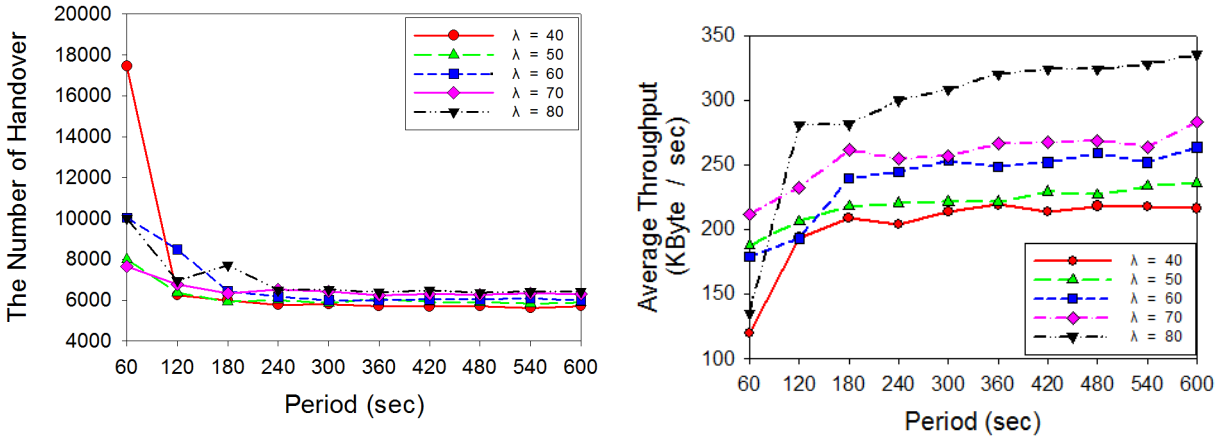
그림 3. 실험 환경
Fig. 3. Simulation enviromment

표 1. 고정된 주기의 알고리즘 실험을 위한 파라미터
Table 1. Simulation parameters for MLB algorithm with static period

Parameters	Value
Bandwidth	$5MHz$ $(25\,PRBs)$
Simulation time	$1{,}800sec$
MLB period	$1 \sim 10min$
The number of UE	100
Mobility model	$Random\ walk\,(3m/s)$
Traffic type	$CBR(64Bytes/0.008sec)$
Duration of bearer	$180sec$
Traffic generation	$Poisson\ random\ variable$ $(\lambda = \{40,50,60,70,80\})$
QoS (max delay)	$0.008sec$
$R_k^{(req)}$	$64Kbps$
ρ_{th}	0
Δ	$0.3dBm$
$O_{max}^{(cn)}$	$3dBm$
Scheduling algorithm	PF algorithm

펨토셀 네트워크를 대상으로 실험하였다.

실험을 위한 펨토셀 네트워크는 그림 3과 같이 7개의 펨토셀로 구성되어 있으며, 각 펨토셀은 공용 접근 모드 (open access mode)로 동작한다. 전체 실험 공간은 $50m \times 47m$ 환경이며, 각 펨토셀간의 거리는 $15m$로 구성되었고, 펨토셀은 반경 $10m$의 서비스 영역을 갖는다. 실험은 총 30분 (1,800초)간 진행이 되며, 총 25개의 PRB (Physical Resource Block)를 갖는 5MHz 대역폭을 사용한다. 사용자 수는 100명이며, 환경 내에서 $3m/s$의 속도로 자유롭게 이동 한다 (random walk). 한편, 하나의 트래픽은 음성 데이터를 고려하여, $64kbps$로 전송을 받으며, 3분 동안 전송이 된다. 트래픽의 양은 포아송 분포 (Poisson distribution)를 따르며, 네트워크의 트래픽 양과 부하 분산의 정도와의 관계를 확인하기 위해, 포아송 분포의 λ값을 다양하게 변화해 가며 실험한다. 그리고 수행되는 주기와 부하 분산의 정도를 확인하기 위해, 수행 주기 또한 다양하게 변화해 가며 실험한다. 각 펨토셀은 PF (Proportional Fair) 스케줄링 알고리즘을 사용하며, QoS (Quality of Service)의 최대 지연 값 (max delay)은 $64kbps$인 것을 고려하여 0.008초로 선택하였다. 기타 식 (1)~(3)에서 사용한 값들은 이전 연구[5]에서 사용한 매크로셀 대상의 변수를 펨토셀에 맞게 수정하였다. 이들 값들은 표 1에 서술하였다. 한편, 실험은 LTE Simulator[7]을 사용하였다. 그리고 실험 시 각 신호 간의 간섭은 편의를 위해 고려하지 않았다. 또한, 중앙 집중식 알고리즘으로 동작하는 주기적인 MLB



(a) (b)

그림 4. 주기의 변화와 네트워크 성능과의 관계 - (a) 주기와 핸드오버 빈도, (b) 주기와 평균 시간당 처리량의 관계
Fig. 4. Relation between period and network performance - (a) relation between period and the number of handover frequency, (b) relation between period and average throughput

알고리즘에서, 인터넷이나 X2 인터페이스를 통한 정보 교환 시 발생하는 지연은 없다고 가정하였으며, 부하 분산 알고리즘이 동작하는 셀은 7개의 셀 중 한 가운데의 셀로 선정하여 실험하였다.

3.2. 실험 결과

먼저 우리는 알고리즘의 주기와 핸드오버의 빈도를 비교하였다. 우선 LTE 네트워크는 하드 핸드오버 (Hard handover)로 동작이 되는데, 이것은 필연적으로 서비스 단절 시간 (Service Interruption Time)을 초래한다[8]. 이 시간은 사용자의 서비스 품질에 밀접한 영향이 있기 때문에, 핸드오버의 빈도를 성능평가의 지표 중 하나로 선택하였다. 그리고 주기의 변화에 따라 이 지표의 값이 어떤 변화를 보이는지 확인하고자 한다. 본 논문에서, 우리는 서비스 단절 시간은 $30ms$으로 정의하였으며, 주기를 변화해 가며 반복적으로 실험을 진행하였다. 그 결과, 그림 4의 (a)와 같이, 핸드오버는 주기가 짧을수록 빈번히 발생하고, 반대로 긴 주기에서는 상대적으로 그 빈도수가 낮았다. 그 이유는, 2절에서 언급한바와 같이, MLB 알고리즘이 수행될 때 마다 펨토셀 외곽의 사용자가 인접한 펨토셀로 강제로 핸드오버가 수행되기 때문이다. 만일 주기가 짧다면, 단위시간동안 MLB 알고리즘의 수행 빈도는 증가할 것이고, 그 결과 강제 핸드오버의 빈도도 증가할 것이다. 따라서 새로운 알고리즘을 제안할 때, 핸드오버의 빈도는 주기의 증가를 통해 달성할 수 있으며, 나아가 서비스 품질의 향상을 기대할 수 있다.

다음으로 우리는 주기의 변화에 따른 펨토셀의 평균 시간당 처리량 (Average throughput)을 분석하였다. 왜냐하면 평균 시간당 처리량은 각 펨토셀의 처리 속도와 효율성을 분석할 수 있기 때문이다. 따라서 우리는 주기의 변화에 따라, 이 지표의 변화 양상을 확인하여, 어느 주기에서 가장 많은 데이터를 단위시간동안 처리했는지 확인하고자 한다. 그 결과는 그림 4의 (b)에서 볼 수 있듯이, 평균 시간당 처리량은 MLB 알고리즘의 수행 주기가 짧을수록 낮았다. 그 이유는 바로 잦은 핸드오버로 인해 서비스 단절 시간이 증가했기 때문이다. 앞서 밝힌 바와 같이, 서비스 단절 시간의 증가는 잦은 핸드오버의 빈도에 기인하는데, 이 시간 동안은 펨토셀과 사용자가 데이터를 전송할 수 없다. 따라서 서비스 단절 시간은 시간당 처리되는 데이터의 양이 줄여, 평균 시간당 처리량의 감소를 유발한다. 만일 새로운 알고리즘을 제안할 때, 평균 시간당 처리량을 증가하고자 한다면, 알고리즘의 주기를 증가시켜서 이를 달성할 수 있다.

한편 위의 두 결과는 주기가 길 때 상대적으로 좋은 성능을 나타낸다. 하지만 이 지표들로 인해 무조건 긴 주기를 선택하는 것은 옳지 않다. 그 이유는 바로 차단율 (Blocking probability) 때문이다. 데이터의 차단은 서비스 제공을 위한 네트워크 용량이 부족해서 발생이 되지만, 전체 네트워크 용량이 충분해도 소수의 펨토셀에서 처리 가능한 용량이 초과되어서 발생할 수 있다. 이 경우, 해당 펨토셀에서 서비스를 받는 사용자 중, 일부의 데이터는 차단이 될 것이다. 따라서 우리는 MLB 알고리즘의 수행 주기와 데이터 차단율의 관계를 확인하고자

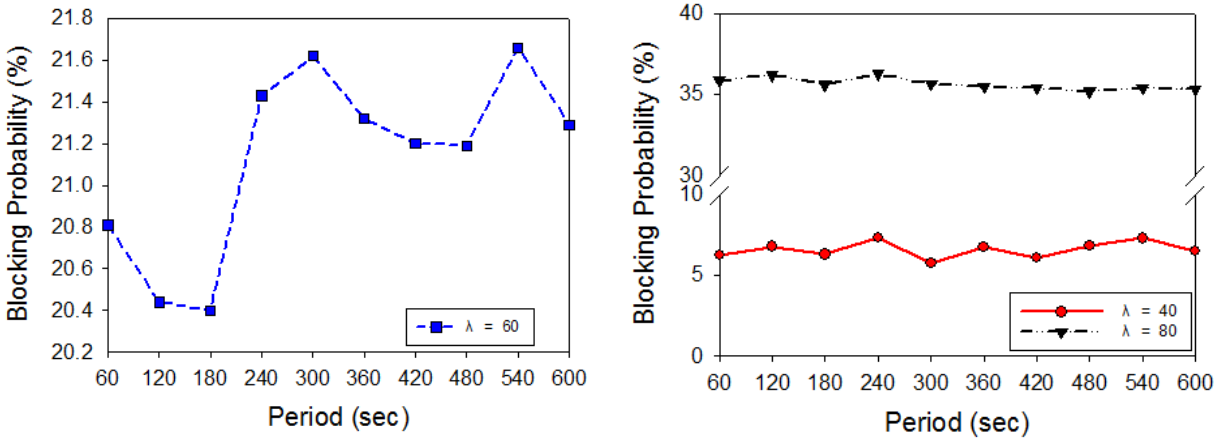


그림 5. 주기의 변화와 데이터 차단율과의 관계 - (a) $\lambda = 60$인 경우, (b) $\lambda = 40$, $\lambda = 80$인 경우
Fig. 5. Relation between period and blocking probability - (a) $\lambda = 60$, (b) $\lambda = 40$, $\lambda = 80$

하며, 그 결과를 그림 5의 그래프로 표현하였다. 먼저 그림 5의 (a)를 보면 알 수 있듯이, 차단율은 주기가 짧아질수록 감소하였다. 이것은 차단율의 감소가 MLB 알고리즘이 자주 수행되어 특정 펨토셀에 집중된 부하를 인접한 셀로 자주 분산시킨 것에 기인한다. 하지만 단순히 이렇게 분석하기에는 무리가 있다. 그림 5의 (b)를 보면 (a)와 달리 뚜렷한 양상이 발견되지 않는데, 그 이유는 네트워크 전체 트래픽의 양 때문이다. λ값이 40인 경우에는, 네트워크 전체 트래픽의 양이 지나치게 적은 경우이다. 이 경우, 차단율의 절대적 수치가 낮고, 그 변화도 크지 않다. 즉 부하가 집중되는 경우가 잘 발생하지 않는다. 반면, λ값이 80인 경우, 즉 네트워크 전체 트래픽의 양이 지나치게 많으면, 그 정도의 차이는 있지만 모든 펨토셀은 높은 부하를 갖는다. 이 경우, 전체의 부하가 네트워크의 처리 가능한 용량에 근접하거나, 그 값을 초과했기 때문에, 부하를 분산해도 인접한 셀에서 처리가 불가능할 수 있다. 그 결과 네트워크 전체 트래픽의 양이 많으면, 부하의 분산을 자주 수행하더라도 차단율은 낮아지지 않는다. 따라서 우리는 새로운 알고리즘을 제안할 때, 차단율을 낮추기 위해서는 먼저 네트워크 전체의 트래픽 양을 고려해야 한다. 전체 네트워크 트래픽의 양이 지나치게 많은 경우에는, 현재 네트워크에 있는 데이터들을 신속히 처리하여 네트워크 전체의 부하를 감소시킨다. 따라서 알고리즘은 네트워크 전체의 부하 감소를 위해, 주기를 최대한으로 증가시켜, 시간당 처리량을 높이도록 동작해야 한다. 반대로 트래픽 양이 낮은 경우에는 차단율 자체가 매우 낮다. 이 경우에는 펨토셀에 부하가 집중되는 경우가 많지 않아, 알고리즘은 차단율 보다 핸드오버의 빈도를 낮추고, 시간당 처리량의 증가를 위해 주기를 늘려야 한다. 하지만 전체 네트워크의 트래픽 양이 많은 경우와는 달리 최대로 늘리지 않는다. 그 이유는, 전체 네트워크의 트래픽의 갑작스러운 증가에 대처하기 위함이다. 마지막으로 앞선 두 경우가 아니라면, 주기를 줄여 차단율을 낮출 수 있다.

우리는 앞선 세 가지의 지표로써 주기에 따른 네트워크의 성능을 비교하였다. 그리고 그 결과를 분석하여 새로운 MLB 알고리즘의 가이드라인을 제시하였다. 이를 종합하면 아래와 같다.

① 핸드오버의 빈도 증가 시: 주기 증가
② 시간당 처리량 감소 시: 주기 증가
③ 데이터 차단율이 상승 시
- 네트워크 내 트래픽 과다: 주기 최대 증가
- 네트워크 내 트래픽 극소: 주기 증가
- 특정 펨토셀에 부하 집중: 주기 감소

우리는 이 결과를 바탕을 다음 장에서 이를 고려한 알고리즘을 제안하고 성능을 분석한다.

Ⅳ. 적응적 (Adaptive) 알고리즘

4.1. 제안 알고리즘

본 장에서는, III장에서 제시한 알고리즘의 가이드라인을 바탕으로 고정된 주기가 아닌, 네트워크의 트래픽 양에 따라 변화하는 적응적 MLB 알고리즘

알고리즘 2. 적응적 주기의 MLB 알고리즘
Algorithm 2. APMLB algorithm

```
Initialize all O_{i,j}^{(cn)} values to 0;
Set initial period of algorithm;
while (every periods) {
    Calculate the ρ_i value for each cell;
    Update O_{i,j}^{(cn)} value for each cell;
    if (traffic ≥ T_th^{(high)})
        period = maxPeriodForHigh;
    else if (traffic ≤ T_th^{(low)})
        period = maxPeriodForLow;
    else
        period = minPeriod;
}
```

(Adaptive Periodic Mobility Load Balancing Algorithm, APMLB)을 제안한다. 한편, 이 알고리즘은 기존의 알고리즘과 마찬가지로, 중앙 집중식 알고리즘이다. 따라서 각 셀의 부하계산 및 핸드오버의 영역 결정은 특정한 하나의 셀에서 수행되고, 그 결과를 각 셀에게 전송한다. 자세한 알고리즘은 아래와 같다.

먼저, 적응적 알고리즘은 기존의 알고리즘과 마찬가지로 $O_{i,j}^{(cn)}$값을 0으로 초기화 한다. 그리고 최초에 동작할 주기를 선정한다. 해당 주기가 되면, 먼저 각 셀들의 부하인 ρ값을 (3)을 이용해 계산한다. 이 후, 계산된 결과를 바탕으로 (2)를 이용하여 핸드오버 영역을 결정한다. 그 이후, 다음에 알고리즘이 수행될 주기를 결정한다. 만일 네트워크의 트래픽 양이 미리 정의된 한계값인 $T_{th}^{(high)}$보다 크다면, 주기를 최대값인 maxPeriodForHigh값으로 설정한다. 반대로, 네트워크의 트래픽 양이 한계값인 $T_{th}^{(low)}$값보다 낮다면, 주기의 값을 미리 설정된 maxPeriodForLow값으로 선정한다. 다시 말해서 네트워크의 트래픽 양이 일정수준보다 크다면, 주기를 maxPeriodForHigh값으로 선정하고, 반대로 일정수준보다 낮다면 maxPeriodForLow값으로 선정한다. 마지막으로, 위의 두 경우가 아닌 경우에는 주기의 값을 가장 낮은 값인 minPeriod값으로 설정한다.

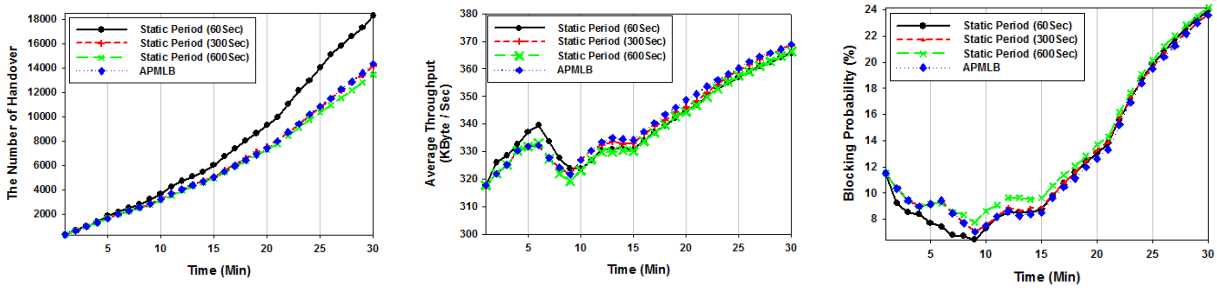
4.2. 실험 환경 및 결과

우리는 본 장에서, 제안된 알고리즘의 성능 평가를 위해 새로운 네트워크 트래픽 환경을 설정한다. 왜냐하면, 제안된 알고리즘은 변화하는 트래픽 환경에서 적합하게 디자인이 되었기 때문이다. 변화하는

표 2. 적응적 알고리즘의 실험을 위한 파라미터
Table 2. Simulation parameters for APMLB algorithm

Parameters	Value
Bandwidth	$5MHz$ $(25\,PRBs)$
Simulation time	$1800sec$
Initial period	$1min$
minPeriod	$1min$
maxPeriodForHigh	$10min$
maxPeriodForLow	$5min$
$T_{th}^{(high)}$	$5.12Mbps$
$T_{th}^{(low)}$	$2.56Mbps$
The number of UE	100
Mobility model	$Random\ walk\,(3m/s)$
Traffic type	$CBR(64Bytes/0.008sec)$
Duration of bearer	$180sec$
Traffic generation $(\lambda, Mbps)$	$0\sim9\,\mathrm{min}:40, 2.56$ $9\sim21\,\mathrm{min}:60, 3.84$ $21\sim30\,\mathrm{min}:80, 5.12$
QoS (max delay)	$0.008sec$
$R_k^{(req)}$	$64Kbps$
ρ_{th}	0
Δ	$0.3dBm$
$O_{max}^{(cn)}$	$3dBm$
Scheduling algorithm	PF algorithm

트래픽 환경은 다음과 같다. 먼저 처음 9분 동안은 네트워크가 낮은 트래픽 양을 갖도록 하였다. 트래픽은 기존 실험과 같이 포아송 분포로 발생을 시키는데, 이 때 포아송 분포의 λ값이 40인 경우를 낮은 트래픽 양으로 선정하였으며, 이는 7개의 펨토셀로 구성된 네트워크에서 초당 5,000개의 56Bytes 패킷을 전송하는 것과 같다 (2.56Mbps). 반대로, 30분 중 마지막 9분 동안은 네트워크 전체가 높은 트래픽 양을 갖도록 하였으며, 이 때의 λ값은 80으로 선정하였다. 이것은 네트워크가 초당 10,000개의 56Bytes 패킷을 전송하는 것과 같다 (5.12Mbps). 그리고 중간의 9분 동안은, 중간값으로 설정하였고, 이는 λ값이 60인 경우와 같다. 마찬가지로 이 값은 네트워크가 초당 7,500개의 56Bytes 패킷을 전송하는 것과 같다 (3.84Mbps). 한편 제안한 알고리즘에서 우리는 초기의 주기 값을 1분으로 선정하였다. 그 이유는, 짧은 시간동안 네트워크의 트래픽 양을 파악하여 적합한 주기를 선정하기 위함이다. 그리고 가장 낮은 주기인 minPeriod값은 1분으로 설정하였다. 또한 네트워크 트래픽 양이 높을 경우, 주기는 10분으로 설정한다 (maxPeriodForHigh=10min). 마지막으로 트래픽 양이 낮은 경우, 주기는 5분으로



(a) (b) (c)

그림 6. 정적인 주기의 알고리즘과 가이드라인이 적용된 알고리즘 (Adaptive) 시간 별 성능 비교 - (a) 핸드오버 빈도수 비교, (b) 평균 시간당 처리량 비교, (c) 차단율 비교

Fig. 6. Changing performance between static and adaptive period MLB algorithm - (a) The number of handover frequency, (b) Average throughput, (c) Blocking probability

설정한다 (maxPeriodForLow=5min). 그리고 알고리즘에서 사용되는 한계값인 $T_{th}^{(high)}$와 $T_{th}^{(low)}$값은 각각 $5.12Mbps$와 $2.56Mbps$의 값으로 선정한다. 이 값은 앞서 정의한 높은 트래픽 양과 낮은 트래픽 양의 값과 같다. 이를 제외한 나머지 실험 환경은 표 1과 동일하다. 위의 서술한 값은 표 2에 서술하였다. 펨토셀의 구성 또한 그림 3과 같이 설정하였다.

위의 환경을 바탕으로 실험하여, 핸드오버의 빈도, 시간당 처리량, 그리고 데이터 차단율의 관점에서 결과를 도출하였고, 그림 6과 7에 나타내었다. 그림 6은 앞서 언급한 세 개의 성능 지표가 시간에 따라 어떻게 변화하는지를 나타낸 결과이고, 그림 7을 실험 시간동안의 전체 성능 지표 값이다.

먼저 우리는 핸드오버 빈도수의 관점에서 정적인 주기의 MLB 알고리즘과, 적응적 주기의 MLB 알고리즘에 대해 비교했다. 그림 6(a)에서 볼 수 있듯이, 먼저 정적인 주기를 갖는 MLB 알고리즘의 경우, 낮은 주기를 갖는 MLB 알고리즘이 높은 주기를 갖는 MLB 알고리즘보다 많은 핸드오버 빈도수를 갖는 것을 알 수 있다. 더불어, 네트워크의 트래픽 양이 많아지는 21분 이후부터는 빈도수의 차이가 점차 심화됨을 알 수 있다. 이것은 II장에서 언급한 바와 같이, 주기가 짧을수록 핸드오버의 빈도수가 높다는 것을 나타낸다. 반면 적응적 주기를 갖는 알고리즘의 경우, 핸드오버의 빈도가 높은 주기를 갖는 MLB 알고리즘와 유사하게 변화함을 알 수 있었다. 그림 7(a)에서도 볼 수 있듯이 전체 실험시간동안 짧은 주기를 갖는 MLB 알고리즘이 이들 중 가장 좋지 못한 성능을 나타내었으며, 긴 주기를 갖는 MLB 알고리즘이 가장 좋은 성능을 내었다. 그리고 적응적 주기의 MLB 알고리즘은 긴 주기를 갖는 MLB 알고리즘과 유사한 성능을 내었다고 할 수 있다. 이것은, 핸드오버 빈도수의 관점에서 볼 때, 높은 주기를 갖는 MLB 알고리즘의 장점을 반영하였다고 할 수 있다.

한편 우리는 평균 시간당 처리량의 관점에서 알고리즘을 비교해 보았다. 비교에 앞서, 우리는 이 그래프를 0분부터 9분까지의 초기구간과 이후의 후기구간으로 나누어서 분석하였다. 먼저 초기구간을 보면, 짧은 주기의 MLB 알고리즘이 가장 높은 값을 나타냄을 알 수 있다. 하지만 II장에서는 주기가 긴 MLB 알고리즘일수록 시간당 처리량이 높다는 그림 6(b)와 반대의 실험 결과를 보였다. 그 이유는 바로 핸드오버 영역이 아직 사용자의 분포에 적합하게 조정되지 않았기 때문이다. 사용자들은 실험 환경에서 언급한 바와 같이 모두 무작위로 배치가 된다. 하지만 실험 시작 당시에는 핸드오버 영역은 모두 동일하여, 사용자의 위치에 따른 영역 조정이 이루어지지 않은 상태이다. 이 상태에서, 짧은 주기를 갖는 MLB 알고리즘의 경우, 빠른 시간 안에 적합한 핸드오버 영역으로 변화하여 적용되기 때문에 다른 주기를 갖는 알고리즘들 보다 좋은 성능을 보인다. 하지만 후기구간에서는, 모든 알고리즘이 사용자의 위치에 따라 핸드오버 영역이 조정된 이후이기 때문에 II장에 결과에 맞게 높은 주기일수록 높은 시간당 처리량을 보인다. 결국 그림 7(b)의 전체 시간동안의 성능으로 보았을 때, 초기구간에서 다소 짧은 주기의 알고리즘이 좋은 성능을 보였다 하더라도 결국은 긴 주기의 알고리즘이 높은 시간당 처리량을 갖는 것을 알 수 있다. 한편 후기구간에서 적응적 주기를 갖는 MLB 알고리즘의 경우, 긴 주기를 갖는 MLB 알고리즘의 결과와 유사함을

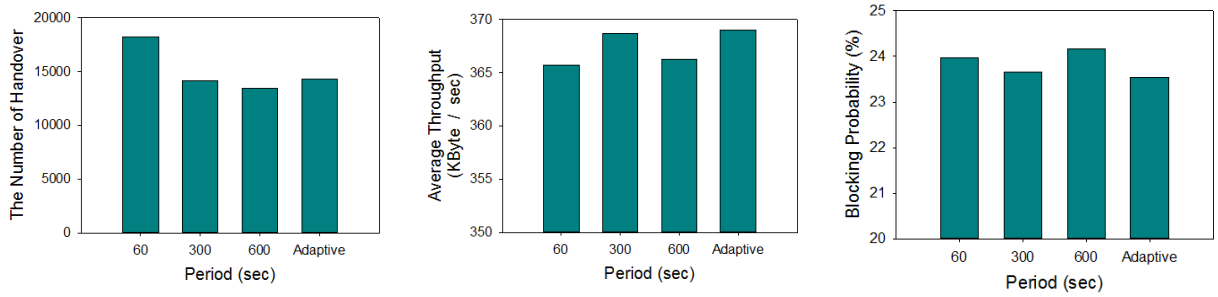


그림 7. 정적인 주기의 알고리즘과 가이드라인이 적용된 알고리즘 (Adaptive) 성능 비교 - (a) 핸드오버 빈도수 비교, (b) 평균 시간당 처리량 비교, (c) 차단율 비교
Fig. 7. Comparison between static and adaptive period MLB algorithm - (a) The number of handover frequency, (b) Average throughput, (c) Blocking probability

볼 수 있다. 이것은 시간당 처리량 관점에서 볼 때, 높은 주기를 갖는 MLB 알고리즘의 장점을 반영하였다고 할 수 있다.

마지막으로 우리는 차단율의 관점에서 알고리즘을 비교하였다. 그림 6(c)에서 보면 알 수 있듯이, 짧은 주기를 갖는 MLB 알고리즘이 긴 주기를 갖는 MLB 알고리즘보다 낮은 차단율을 갖는 것을 알 수 있다. 한편 적응적 주기를 갖는 MLB 알고리즘의 경우에는 초기구간인 9분까지는 짧은 주기를 갖는 MLB알고리즘보다 차단율이 높다. 그 이유는 앞서 시간당 처리량을 분석할 때와 마찬가지로, 사용자의 위치에 맞게 핸드오버 영역이 아직 조정되지 않았기 때문이다. 하지만 후기구간으로 갈수록, 차단율은 점차 짧은 주기를 갖는 MLB 알고리즘과 유사한 값을 갖게 된다. 전체적인 차단율의 성능은, 그림 7(c)에서 볼 수 있듯이 짧은 주기를 갖는 MLB 알고리즘과 유사한 값을 갖는 것을 알 수 있다. 따라서 적응적 주기의 MLB 알고리즘은 짧은 주기를 갖는 MLB 알고리즘의 장점을 반영한 알고리즘이라고 할 수 있다.

위의 세 결과로 미루어 보았을 때, 적응적 주기를 갖는 MLB 알고리즘은 짧은 주기를 갖는 MLB 알고리즘과, 긴 주기를 갖는 MLB 알고리즘의 장점을 반영하여 상황에 맞게 적응적으로 변화함을 알 수 있다.

V. 결 론

본 논문에서는 펨토셀 네트워크 상에서, 기존에 제안된 주기적인 MLB 알고리즘의 성능 변화를 분석하고, 이 정보를 바탕으로 적응적 주기의 MLB 알고리즘을 제안하였다. 기존의 주기적인 MLB 알고리즘의 경우, 네트워크 트래픽 양을 고려하지 않고 고정된 주기로 동작을 하여, 자주 네트워크 트래픽 환경이 변화하는 펨토셀 네트워크에서는 적합하지 않다. 따라서 우리는 네트워크 환경에 따라 주기를 변화시켜, 처해진 상황에 맞게 적응적으로 주기를 바꿔 동작하는, 적응적 주기의 MLB 알고리즘을 제안하였다.

향후 우리는 CR (Cognitive Radio) 기술이 적용된 펨토셀 (Cognitive Femtocell) 네트워크 상에서의 부하 분산 알고리즘을 연구하고자 한다. CR 기술이 적용된 펨토셀은 일반적으로 셀 간의 간섭을 완화하기 위해 제안되었다. 하지만 우리는 CR 기술을 통하여 사용자 단말의 위치를 인지해 부하를 분산함과 동시에, 셀 간의 간섭 또한 고려한 부하 분산 알고리즘의 연구를 진행하고자 한다.

References

[1] V. Chandrasekhar, J. Andrews, and A. Gatherer, “Femtocell networks: a survey,” *IEEE Commun. Mag.*, vol. 46, no. 9, pp. 59-67, Sep. 2008.

[2] S. Yeh, S. Talwar, S. C. Lee, and H. Kim, “WiMAX femtocells: a perspective on network architecture, capacity, and coverage,” *IEEE Commun. Mag.*, vol. 46, no. 10, pp. 58-65, Oct. 2008.

[3] O. G. Aliu, A. Imran, M. A. Imran, and B. Evans, “A survey of self organization in future cellular networks,” *IEEE Commun. Surveys*

Tutorials, vol. 15, no. 1, pp. 336-361, First Quarter 2013.

[4] A. Lobinger, S. Stefanski, T. Jansen, and I. Balan, "Load balancing in downlink LTE self-optimizing networks," in *Proc. IEEE Veh. Technol. Conf. Spring (VTC 2010-Spring)*, pp. 1-5, Taipei, Taiwan, May 2010.

[5] R. Kwan, R. Arnott, R. Paterson, R. Trivisonno, and R. Kubota, "On mobility load balancing for LTE systems," in *Proc. IEEE Veh. Technol. Conf. Fall (VTC 2010-Fall)*, pp. 1-5, Ottawa, Canada, Sep. 2010.

[6] W. Kim, J. Y. Lee, and Y. J. Suh, "A study on the periodic mobility load balancing algorithm in LTE femtocell networks," in *Proc. KICS Winter Conf. 2013*, pp. 246-247, Yongpyeong, Korea, Jan. 2013.

[7] G. Piro, L. A. Grieco, G. Boggia, F. Capozzi, and P. Camarda, "Simulating LTE cellular systems: all open-source frameworks," *IEEE Trans. Veh. Technol.*, vol. 60, no. 2, pp. 498-513, Oct. 2010.

[8] K. Dimou, M. Wang, Y. Yang, M. Kazmi, A. Larmo, J. Pettersson, W. Muller, and Y. Timner, "Handover within 3GPP LTE: design principles and performance," in *Proc. IEEE Veh. Technol. Conf. Fall (VTC 2009-Fall)*, pp. 1-5, Anchorage, U.S.A., Sep. 2009.

김 우 중 (Woojoong Kim)

2012년 2월 홍익대학교 컴퓨터공학과 학사

2012년 3월~현재 포항공과대학교 컴퓨터공학과 통합과정

<관심분야> 셀룰러 네트워크, LTE, 펨토셀

이 정 윤 (Jeong-Yoon Lee)

2006년 2월 동국대학교 컴퓨터공학과 학사

2013년 8월 포항공과대학교 컴퓨터공학과 박사

<관심분야> 애드혹 네트워크, 무선랜, 무선네트워크

서 영 주 (Young-Joo Suh)

1985년 2월 한양대학교 전자공학과 학사

1987년 2월 한양대학교 전자공학과 석사

1996년 미국 조지아 공대 (Georgia Tech) 컴퓨터공학 박사

1988년~1990년 LG전자 연구원

1990년~1993년 충청대학 교수

1996년~1997년 미국 Georgia Tech. 연구원

1997년~1998 년 미국 University of Michigan 연구원

1998년~현재 포항공대 컴퓨터공학과 교수

<관심분야> 무선랜 프로토콜, 이동 IP, 이동 멀티캐스트, ad-hoc/sensor 네트워크, 차세대 이동 네트워크