韓國經營科學會誌
第26卷 第2號
2001年 6月

광마이크로셀 이동통신 시스템의 균등부하를 위한 셀단위 핸드오프 순서결정

이채영* · 장세헌**

Minimization of Cell-based Handoff Delays to Balance the Load in Fiber Optic Micro-cellular Systems

Chae Y. Lee* · Se H. Jang**

■ Abstract ■

This paper considers the scheduling of cell-based handoffs to balance the traffic in a fiber-optic microcellular system. In the system depending on the order of cell based handoff, periodical balancing of the traffic among microcells can be achieved.

The cell based handoff problem is formulated as a dynamic programming and the computational complexity is analyzed. Since the scheduling problem requires real time solution, heuristic algorithms are proposed and the computational results are discussed.

Keyword : microcellular system, handoff, load balancing, scheduling, dynamic programming, heuristic algorithm

1. 서 론

마이크로셀은 기존의 셀에 비하여, 송신 비용을 경감하고, 시스템의 용량을 늘릴 수 있다. 또한 시스템의 설치가 보다 용이하기 때문에, 음영지역을 감소시킬 수도 있다. 그래서, 보다 다양한 서비스

논문접수일 : 2000년 1월 18일 논문게재확정일 : 2001년 3월 8일
* 한국과학기술원 산업공학과 교수
** 한국과학기술원 산업공학과

를 제공하고, 높은 QoS(Quality of Service) 를 보장하게 된다.

광 마이크로 셀룰러 시스템에서 몇 개의 마이크로셀(mBS)은 섹터로 묶여 지고, 몇 개의 섹터는 다시 중앙국의 가상기지국으로 묶여 지게 된다[1]. 그래서, 계층적 구조를 갖게 된다. 그러나, 마이크로셀이 속해있는 섹터가 고정되어 있다면, 호통화량은 유동적으로 변하기 때문에 시스템의 용량을 충분하게 크게 해야 사용자들의 요구를 만족시킬 수 있을 것이다. 따라서, 마이크로셀이 상황에 따라 유연하게 각 섹터에 배치되게 한다면, 시스템의 자원을 보다 효율적으로 사용할 수 있다[2, 5]. 이런 마이크로셀은 중앙국의 OMC(Operation Management Center)에서 Pilot PN Offset을 변화시킴으로써 제어된다[4]. 즉, 각각의 마이크로셀은 상황에 따라 자신이 속한 섹터와 가상기지국이 변경될 수 있다. 가상기지국에서 마이크로셀까지는 광섬유로 연결되어 있고, 마이크로셀에서 이동국까지는 무선환경이다.

이 시스템에서 섹터와 가상기지국에는 각각 채널자원이 한정되어 있기 때문에, 특정한 섹터와 가상기지국에 호통화량이 밀집되어 있다면, 이동국들의 통화품질이 저하되고 호불통율 또한 높아지게 된다. 다시 말하면 CDMA(Code Division Multiple Access) 방식에서는 채널간의 간섭(interference)보다 해당 기지국, 또는 해당 섹터 내에서의 이동국의 수에 따라 또한 각 이동국의 출력제어 방법에 따라 noise level이 결정되기 때문에, 섹터간의 호통화량을 균등 분배함이 중요한 문제가 된다. 따라서, 일정한 시간 간격으로 마이크로셀을 재배치함으로써 호통화량을 섹터와 가상기지국마다 균등하게 구성할 필요가 있다[6]. 셀을 어떻게 구성하여 섹터화하는가는 그룹핑문제에 해당한다. 그러나, 통화량의 변화에 따른 재할당을 위해 마이크로셀을 다른 섹터로 옮기는 과정에서 핸드오프가 발생하게 되고, 셀이 재배치되는 과정에서 핸드오프를 어떠한 순서로 해야 하는지에 대한 문제가 발생한다.

섹터와 가상기지국의 채널자원이 제약조건이기 때문에 마이크로셀의 핸드오프 순서에 따라서 현재 섹터화 상태에서 다음 섹터화 상태로 마이크로셀들의 핸드오프가 가능할 수도 있고 가능하지 않을 수도 있게 된다. 또한 마이크로셀의 재배치는 주기적으로 발생하게 되기 때문에, 셀을 핸드오프시키는 것도 실시간적으로 처리되어야 한다.

이 논문에서 다루어지는 핸드오프는 강제 핸드오프(forced handoffs) 에 해당한다. 흔히 다루어지는 핸드오프는 이동국이 자신이 속한 셀에서 다른 셀로 이동함으로 발생하지만, 광마이크로 셀룰러 시스템에서 섹터간 균등 부하를 위한 핸드오프는 이동국이 이동하지 않음에도 불구하고 시스템의 한정된 채널자원을 효율적으로 사용하기 위해서 한 섹터에 편중된 호통화량을 다른 섹터에 분산시키기 위해 하나의 마이크로셀에 속한 호통화량 모두를 다른 섹터로 강제 핸드오프 시킨다.

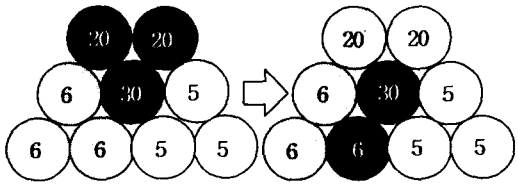
2절에서는 이 문제를 보다 구체화시키고, 3절에서는 동적계획법(Dynamic Programming)으로 문제를 정식화한다. 4절에서는 실시간적으로 문제를 해결할 수 있는 휴리스틱 알고리즘을 제시한다. 5절에서는 시뮬레이션 결과를 비교 분석해 본다.

2. 셀단위 핸드오프의 순서결정

광 마이크로 셀룰러 시스템은 서론에서 이야기한 것처럼 마이크로셀, 몇 개의 마이크로셀을 포함하는 섹터(sector), 그리고 몇 개의 섹터를 포함하는 가상기지국(virtual base station)으로 구성된다. 이 시스템에서 각각의 섹터와 가상기지국은 한정된 채널자원을 갖고 있다. 그래서 호불통율(call blocking)이나 호절단율(call dropping)을 줄이고 보다 나은 통화 품질을 유지하기 위해서, 각각의 마이크로셀은 자신이 속한 섹터나 가상기지국을 변경하여 선택하게 된다. 이렇게 섹터와 가상기지국을 변경하여 선택하는 것은 마이크로셀의 입장에서는 자기가 속한 각 호의 핸드오프에 해당한다. 이러한 핸드오프는 이동국(mobile)이 지금 서비스를 받고 있는 마이크로셀의 서비스지역을 벗어나서 발생하는 핸드오프가 아니라 호불통율이나 호절단율

또는 통화품질을 위해서 이동국의 서비스지역 이탈에 관계없이 강제적으로 발생하는 핸드오프이다.

섹터나 가상기지국의 채널자원은 한정되어 있다. 따라서 마이크로셀의 현재 섹터 상태와 다음 섹터 상태가 주어져 있을 때 핸드오프시키는 마이크로셀이 이동해 가려는 섹터와 가상기지국의 채널자원을 초과하지 않아야 하기 때문에 마이크로셀을 어떠한 순서로 이동시키는가에 따라 강제 핸드오프가 가능할 수도 있고 불가능할 수도 있다.



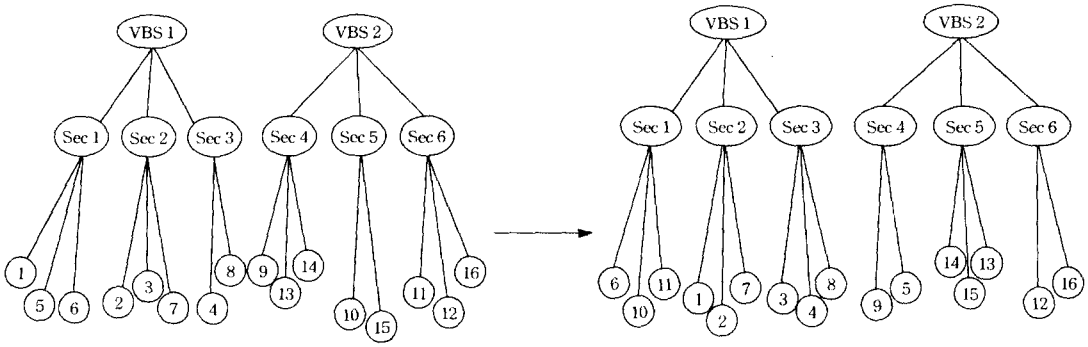
Current Sectorization Next Sectorization

[그림 1] 마이크로셀의 균등분하

[그림 1]에서 같은 무늬를 가진 원이 같은 섹터에 속한 마이크로셀을 나타내고 있다. 각각의 섹터는 섹터 당 40채널의 한정된 자원을 갖는다고 가정할 때, 현재 섹터 구성으로는 위에 위치한 섹터에서 30채널자원의 초과분이 발생하여 호절단이 발생하게 된다. 그러나, 섹터 구성을 오른쪽과 같이 변경하게 되면 모든 섹터에서 각각 채널자원을 초과하지 않게 되고, 호절단률이나 호불통률이 낮아지게 된다. 이러한 이유로 마이크로셀의 균등분하가 필요하게 된다.

[그림 2]의 (a)는 현재의 마이크로셀이 어떠한 섹터와 가상기지국에 속해있는지를 보여주고 있다. 이러한 상황에서 어느 한 섹터에 호통화량이 과도하게 부과되면, (b)와 같은 마이크로셀의 재배치가 필요하다. [그림 2]의 (a), (b)를 비교해 보면, 섹터 1에 속해있는 마이크로셀 1, 5가 섹터 2, 4로 각각 핸드오프된 것이다. 이렇게 마이크로셀의 섹터구성을 바꿔 주어야 할 필요가 발생할 때, 실제로 섹터 구성이 바뀌는 과정은 한 마이크로셀을 현재 속한 섹터에서 다른 섹터로 핸드오프 시키는 것이다. 핸드오프 시켜야 할 마이크로셀이 여러 개 발생할 경우에 각각의 섹터는 채널자원이 한정되어 있기 때문에, 어떤 마이크로셀부터 핸드오프 시킬지 결정하는 문제가 발생하게 된다. 마이크로셀이 핸드오프 되어야 하는 상황을 그래프로 나타내면 핸드오프 순서결정이 발생하는 이유를 보다 쉽게 설명할 수 있다.

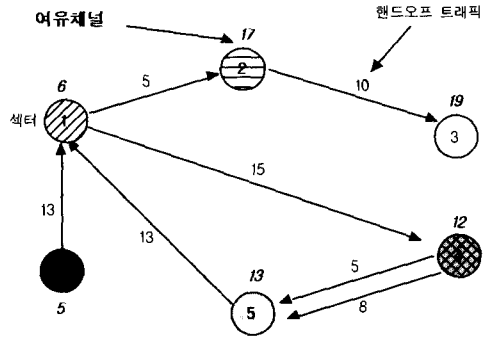
[그림 3]에서 노드는 섹터를 뜻하고, 노드에서 노드로 방향성을 가진 아크는 핸드오프 되는 마이크로셀을 나타낸다. [그림 3]에서 노드 1에서 노드2로 표시된 아크는 섹터 1에서 섹터 2로 핸드오프되



(a) 현재 섹터 구성 (b) 요구되는 섹터 구성

[그림 2] 마이크로셀의 계층구조

는 마이크로셀 1번을 뜻하고, 아크 위에 표시된 5라는 숫자는 마이크로셀 1이 갖고 있는 호통화량이 5 라는 것을 의미한다. 그리고, 노드의 위에 있는 값들은 채널자원에서 섹터가 가진 호통화량의 총합을 뺀 것으로 섹터의 채널자원의 여유분을 나타낸다.



[그림 3] 그래프로 표시된 마이크로셀의 핸드오프 상황

예를 들어, 지금 상태에서는 섹터 1에서 섹터 4로 핸드오프 되어야 하는 15 통화량을 가진 마이크로셀이 섹터 4의 채널자원의 부족으로 핸드오프가 불가능하다. 그러나, 섹터 4에서 섹터 5로 핸드오프 되어야 하는 5와 8의 통화량을 가진 두 마이크로셀이 핸드오프 되면, 섹터 4는 여유채널이 12에서 25로 증가하기 때문에, 그전에 핸드오프가 불가능했던 15 통화량을 가진 마이크로셀이 섹터 1에서 섹터 4로 핸드오프가 가능하게 된다. 이렇게 핸드오프 되어야 하는 마이크로셀들에 대해서, 그 핸드오프 순서에 따라 모든 핸드오프를 필요로 하는 마이크로셀을 핸드오프 시킬 수도 있고, 그렇지 못할 수도 있다. 또한 채널자원을 초과하지 않는다면 한 단계에서 여러 마이크로셀을 동시에 핸드오프 시키는 것이 전체적인 핸드오프 시간을 줄일 수 있게 된다. [그림 3]에서는 첫 번째 단계에서 모두 4개의 마이크로셀이 각 섹터의 채널자원을 초과하지 않고 핸드오프가 가능하다.

이 문제는 마이크로셀을 어떠한 순서로 핸드오프 시킬지를 결정해야 하는 문제이기 때문에, 가능한 해가 있다면 그것을 찾고 해들 중에서 가장 적은 단계로 모든 셀을 핸드오프 시킬 수 있는 순서를 결정하는 것이 목적이 된다.

3. 동적계획법을 이용한 알고리즘

동적계획법(Dynamic Programming)은 다단계 의사결정 문제의 최적화 기법으로 전체 문제를 여러 단계의 작은 문제로 분할한 후 작은 문제들간의 상호관계를 순환식으로 표시하여 이들 순환식을 단계적으로 풀어나감으로써 전체 문제의 최적해를 구하는 기법이다[3, 7]. 다음 식 (1), (2) 에서 핸드오프를 필요로 하는 모든 마이크로셀들을 몇 단계에 걸쳐 핸드오프 시킬 수 있는지 상호관계를 나타내는 순환식을 보여주고 있다.

$$f_n(S_n, X_n) = \min_{Y_n} \{1 + f_{n+1}, (S_{n+1}, X_n - Y_n)\}$$
$$n = 1, \cdots, N-1 \quad (1)$$
$$f_N(S_N, \phi) = 0 \quad (2)$$

S_n : 단계 n에서의 섹터 여유채널
X_n : 단계 n에서의 핸드오프가 필요한 마이크로셀의 집합
Y_n : 단계 n에서의 핸드오프된 마이크로셀의 집합
f_n : 단계 n까지 마이크로셀을 핸드오프 시키는데 필요한 최소회수

식 (1)과 (2)는 동적계획법의 정식화를 나타내고 있다. X_n는 핸드오프 시켜야 하는 마이크로셀의 집합을 나타내고, Y_n는 X_n의 공집합을 제외한 한 부분집합이다.

동적계획법에서는 단계(stage), 상태(state), 입력변수(input variable), 결정 변수(decision variable), 그리고 수익(return) 및 다음 단계로의 변환(transition)으로 문제를 표현된다. f_n는 단계 n까지 마이크로셀을 핸드오프 시키는데 필요한 최소회수를 의미한다. 그리고, 각 단계 n에서 남은 마이크로셀 중에서 강제 핸드오프 하려는 마이크로셀을 나타

내는 Y_n가 결정되면 그때 다음 단계에서의 상태 $X_n - Y_n$이 결정된다. 보통의 동적계획법에서는 역방향으로 푸는 마지막 단계에서 목적함수의 최적값이 결정된다. 이 문제에서는 각 단계의 수익이 1이 되어 마지막 단계에서의 f값이 모든 마이크로셀을 핸드오프 시키는데 필요한 최소 회수가 되며 그 값이 N과 같다.

위의 식 (1)에서 f의 변수 $X_n - Y_n$은 아직 핸드오프 되지 않은 마이크로셀의 집합을 나타낸다. 그래서, 다음 순환식에서 $X_n - Y_n$의 부분집합인 모든 Y_n에 대해여 마이크로셀을 핸드오프시키는 것이 가능한지를 확인하고, 다시 $X_n - Y_n$은 부프로그램의 변수가 된다. 그리고, 실현가능한 해인지 아닌지는 섹터의 여유채널을 가지고 판단할 수 있고, S_n 또한 S_{n+1}으로 값을 변화시켜 다음 부프로그램의 변수값이 된다. 이렇게 자기호출을 반복하여 X_n의 개수를 계속 줄여나가게 되고, X_n이 공집합이 되면, 그 때의 f_n값이 모든 셀을 핸드오프시키는 가장 최소회수가 된다. 만약 X_n이 공집합이 되기 전에 S_n의 값에 따라 핸드오프시킬 수 있는 마이크로셀이 없다면, 그 문제는 실현불가능한 해이다.

그러나, 동적계획법을 이용한 알고리즘은 최적해를 제공하지만 많은 메모리와 계산시간을 필요로 한다. X_1의 원소개수 m에 따라 최대 몇 개의 부프로그램을 호출해야 하는 지를 계산해 보면, 최소한 2^m 이상이 되는 것을 알 수 있다.

U_m을 X_1에 대한 순환식 (1)의 계산회수라고 하자. 즉 U_m은 핸드오프 시켜야 하는 마이크로셀의 개수가 m일 때 동적계획법에서 최대로 호출할 수 있는 부프로그램의 개수이다. 부프로그램을 호출하여 계산하는 시간이 일정하다고 가정을 한다면, 동적계획법의 계산 복잡도는 아래와 같이 표현된다.

$$U_m = \sum_{k=1}^{m} C_k U_{m-k} \tag{3}$$

식 (3)에서 $U_{m-1}, U_{m-2}, U_{m-3}, \cdots, U_1, U_0$을 모두 1이라고 가정하면, 이항정리에 의해서 U_m은 2^m이 된다. 따라서, 동적계획법의 계산복잡도는 최소한 2^m 이상이 되기 대문에 동적계획법을 이용한 알고리즘으로는 실시간적으로 계산이 불가능하다.

실제로 프로그램이 구현되는 역방향 해결법에 따라서 계산복잡도를 구해보면, $m = |X_1|$이라고 위와 같이 정의하고, 부분집합 Y_n가 한번에 핸드오프 가능한지 아닌지를 검사하는 것을 하나의 단위로 생각해 보자.

우선 X_n의 모든 부분집합 Y_n에 대해서, 한번에 이동 가능한지를 체크 하는 것이 역방향 해결법에 의한 첫 번째 방법이다. 이러한 것이 계산 복잡도의 첫 번째 부분 2^m을 나타낸다.

이 문제를 풀 때 Y_n의 마이크로셀의 개수가 가장 작은 1부터 m까지 차례로 늘리면서 풀어나가는 방법을 택하였기 때문에, $i = |Y_n|$라고 가정할 때, i 만큼 마이크로셀을 가진 Y_n는 ${}_mC_i$만큼 존재하게 되고, 또 Y_n의 모든 부분 집합에 대해서 그 부분집합들이 한번에 핸드오프 가능한지 아닌지를 검사해 봐야 하기 때문에, 2^i만큼 부분집합의 개수가 존재한다. 그래서, 실제로 $\sum_{i=1}^{m} 2^i {}_mC_i$ 만큼의 검사를 최소회수를 찾을 때 행하여야 한다.

그래서, 최종 계산 복잡도는 다음과 같이 표현할 수 있다.

$$O(2^m + \sum_{i=1}^{m} 2^i {}_mC_i) \tag{4}$$

그런데, $\sum_{i=1}^{m} 2^i {}_mC_i$은 이항정리에 의해서 $\sum_{i=0}^{m} 2^i {}_mC_i - 1 = (2+1)^m - 1$로 표현되어 그 복잡도는 다음과 같다.

$$O(2^m + 3^m) \tag{5}$$

따라서 동적계획법은 핸드오프 시켜야 하는 마이크로셀의 개수 m에 대해여 지수적으로 증가하는 계산복잡도를 나타낸다.

4. 휴리스틱 알고리즘

3절에서 동적계획법을 이용하여 최적해를 구하는 알고리즘을 제시하였지만 동적계획법의 계산복잡도는 마이크로셀의 개수가 증가함에 따라 지수적으로 증가하기 때문에 실제 적용시키기에는 어려움이 있다. 따라서 문제의 특성을 고려하여 핸드오프시키고자 하는 마이크로셀 중에서 쉽게 핸드오프가 가능한 것을 먼저 핸드오프시킴으로 해서 전체적으로 선택하고자 하는 마이크로셀은 줄여서 문제를 풀어 나가는 휴리스틱 알고리즘을 제시하고자 한다. 예를 들어, 한 섹터의 여유채널이 그 섹터로 핸드오프되어 오는 마이크로셀들의 트래픽의 합보다 크다면 그 섹터로 핸드오프되어 오는 모든 마이크로셀들을 동시에 핸드오프 시키는 것이 가능하다. 만약 그러한 섹터가 여러 개 있어도 그 섹터들로 핸드오프 되어 오는 마이크로셀들은 동시에 핸드오프가 가능하다. 따라서, 이러한 조건을 만족시키는 마이크로셀들을 처음에 찾아서 핸드오프 시켜 주면, 핸드오프 해야 하는 마이크로셀의 총개수는 쉽게 줄일 수 있다.

여기서 알고리즘의 각 단계에서 핸드오프가 필요한 마이크로셀 x에 대하여 x를 핸드오프 시킨 후에 추가적으로 핸드오프 시킬 수 있는 마이크로셀의 트래픽의 총합을 $g(x)$라 하면 다음과 같은 알고리즘을 제안할 수 있다.

Step 1. 현재 섹터, 다음 섹터 그리고 트래픽을 갖는 마이크로셀들을 입력받고, 변수들을 초기화 시킨다.

Step 2. 각 섹터로 핸드오프되어야 하는 마이크로셀이 동시에 핸드오프가 가능한지를 검사하고, 가능하다면 모두 핸드오프 시킨다.

Step 3. Step 2에서 핸드오프 시킨 마이크로셀이 존재한다면, 다시 Step 2로 간다.

Step 4. 핸드오프 가능한 마이크로셀 x중에서 $g(x)$가 가장 큰 x를 선택하여 핸드오프 시킨다.

Step 5. Step 4에서 핸드오프 시킨 마이크로셀이 존재한다면, 다시 Step 2로 간다.

Step 6. 모든 마이크로셀이 핸드오프 되었다면 해를 출력하고, 그렇지 않다면 본 알고리즘으로 실현불가능이라고 출력한다.

이 논문에서 제시하는 휴리스틱 알고리즘은 문제의 특성을 이용하여 한번에 핸드오프가 가능한 마이크로셀은 핸드오프를 해 나가면서 핸드오프 순서를 결정한다. 그래서, 어떤 마이크로셀을 핸드오프 시킬지 결정하는 기준으로 $g(x)$를 사용한다. $g(x)$가 갖는 의미는 이 마이크로셀을 핸드오프 시켰을 때에 추가로 핸드오프 시킬 수 있는 마이크로셀의 트래픽 크기를 비교하여, 핸드오프 시켰을 때에 다른 마이크로셀이 핸드오프 될 수 있는 여유채널을 더 많이 만들어 주는 마이크로셀을 핸드오프 시키자는 것이다.

Step 2에서는 동시에 핸드오프 가능한 마이크로셀들이 있는 지를 검사하고, 각 섹터의 채널자원을 초과하지 않으면서 동시에 핸드오프 가능한 셀들을 찾아서 핸드오프 시킨다. 이러한 과정을 통해서 어느 정도 채널 자원의 여유가 있다면 많은 마이크로셀들을 핸드오프 시킬 수 있다. 그러나, 전체적으로 호통화량이 아주 많은 시간대에는 각각의 마이크로셀들이 여유채널이 크지 않기 때문에, 여러 마이크로셀을 동시에 옮기는 것이 가능하지 않게 되고, 그러한 경우에는 각각의 모든 마이크로셀들을 비교해 보고 핸드오프 순서를 결정해야 한다. 그러한 과정을 담당하는 부분이 Step 4이다.

Step 4에서는 위에서 언급했듯이 $g(x)$라는 값을 도입하여 기준으로 삼고, 각각의 마이크로셀의 순서결정에 이용하고 있다. 그래서, 아래의 두 가

지 방법을 고려할 수 있는 바, 그에 따라 휴리스틱 알고리즘 1, 2로 구분한다.

- Algorithm 1
  - 전체 시스템에서 모든 핸드오프 가능한 마이크로셀 중에서 x가 가장 큰 $g(x)$를 선택하여 핸드오프 시킨다.
- Algorithm 2
  - 각각의 섹터에서 자신에게 핸드오프 해 올 수 있는 마이크로셀 x들 중에서 $g(x)$가 가장 큰 x를 선택하여 동시에 핸드오프 시킨다.

Algorithm 1에서는 핸드오프 시켜야 하는 모든 마이크로셀 중에서 $g(x)$가 가장 큰 마이크로셀을 찾고 핸드오프 시킨다. 핸드오프 시켰을 경우에 자신이 속해 있던 섹터의 여유채널은 증가하게 되기 때문에, 그로 인하여 다른 마이크로셀들의 불가능했던 핸드오프를 가능하게 만들어 줄 수 있게 된다.

Algorithm 2는 첫 번째 알고리즘에서 전체적으로 $g(x)$가 가장 큰 마이크로셀 하나를 선택하는 것에 비하여, 각각의 섹터로 핸드오프 해야 하는 마이크로셀 중에서 $g(x)$가 가장 큰 마이크로셀을 선택하게 된다. 즉, Algorithm 1에서는 Step 4에서 하나의 마이크로셀만 선택되는 반면, Algorithm 2에서는 최대 섹터 수만큼 마이크로셀이 선택될 수 있다. 또한 그러한 마이크로셀들은 목적 섹터가 서로 다르다. 그래서 각각 핸드오프 가능한 마이크로셀들이기에 각 섹터에서 $g(x)$가 가장 큰 마이크로셀을 선택하여 동시에 핸드오프 시키는 것이 가능하다. 그래서, 두 번째 알고리즘이 첫 번째 알고리즘에 비해서 문제가 복잡하게 되었을 경우에 보다 마이크로셀의 핸드오프 회수를 줄일 수 있게 된다.

하나의 마이크로셀이 핸드오프 가능한지의 여부를 판단하는 것을 한 단위로 할 때, 각 알고리즘의 복잡도는 다음과 같다.

$$\text{Algorithm 1} : O(m^3)$$
$$\text{Algorithm 2} : O(2m^3)$$

Algorithm 1에서 Step 2의 경우에 해당하는 핸드오프는 없고, Step 4의 경우만 발생한다고 할 때, 핸드오프 해야 하는 마이크로셀의 개수 m의 제곱만큼 검색하게 되고, $g(x)$를 모든 마이크로셀에 대하여 검색하기 때문에 계산복잡도는 위와 같다. 즉, 핸드오프 시켜야 하는 마이크로셀을 결정하는데 m의 검색이 필요하다. 또한 핸드오프 시켜야 하는 마이크로셀이 m개 존재한다. 그래서 m^2만큼 마이크로셀의 검색이 발생한다. 그런데 핸드오프 가능한 마이크로셀이 여러 개 일 때, 그 중에서 $g(x)$가 큰 값을 결정해야 한다. 이 핸드오프 가능한 마이크로셀도 또한 최대 m만큼 가능하게 된다. 그래서 Algorithm 1의 계산 복잡도는 $O(m^3)$가 된다.

Algorithm 2에서 m^2 만큼의 마이크로셀 검색이 필요한 것은 Algorithm 1과 동일하다. 그러나 $g(x)$를 계산할 때 각각의 섹터에서 값이 가장 큰 마이크로셀을 찾아야 하기 때문에 Algorithm 1에 비하여 함수 하나가 추가가 되었다. 그 함수는 모든 섹터에서 $g(x)$가 가장 큰 마이크로셀 를 찾아서 기억하는 역할을 한다. 그래서 $g(x)$를 찾는 부분이 m을 그리고 각 섹터당 가장 큰 $g(x)$를 찾는 부분이 또 m만큼 검색이 필요하다. 따라서 Algorithm 2의 계산 복잡도는 $O(m^2(m+m))$이 되어서 $O(2m^3)$가 된다.

제안된 휴리스틱 알고리즘을 동적계획법과 비교할 때 마이크로셀의 개수에 대해서, 동적계획법의 계산복잡도는 지수함수로 나타나지만 휴리스틱 알고리즘 1, 2는 모두 다항함수로 표현된다. 따라서, 휴리스틱 알고리즘의 계산 복잡도가 훨씬 낮게 나타난다.

5. 실험결과

동적계획법을 이용한 방법과 Algorithm 1, 2를 네가지 환경 1) 12 cells, 3 sector, 1 VBS, 2) 19 cells, 6 sector, 2 VBS, 3) 37 cells, 9 sector, 3 VBS, 4) 61cell, 12 sectors, 4 VBS에서 각각 실험하

〈표 1〉 12-마이크로셀에 대한 동적계획법과 휴리스틱 알고리즘의 실험 결과 비교

Problem	# of microcells to handoff	DP # of steps	DP CPU time	Heuristic 1 # of steps	Heuristic 1 CPU time	Heuristic 2 # of steps	Heuristic 2 CPU time
1	5	2	0.11	2	0.00	2	0.11
2	4	2	0.05	2	0.00	2	0.00
3	4	1	0.00	1	0.00	1	0.00
4	4	1	0.00	1	0.00	1	0.00
5	2	1	0.00	1	0.00	1	0.00
6	3	1	0.00	1	0.00	1	0.06
7	3	1	0.00	1	0.00	1	0.00
8	4	1	0.00	1	0.00	1	0.00
9	3	1	0.00	1	0.00	1	0.00
10	3	1	0.00	1	0.00	1	0.00
평균	3.5		0.02		0.00		0.02

〈표 2〉 19-마이크로셀에 대한 동적계획법과 휴리스틱 알고리즘의 실험 결과 비교

Problem	# of microcells to handoff	DP # of steps	DP CPU time	Heuristic 1 # of steps	Heuristic 1 CPU time	Heuristic 2 # of steps	Heuristic 2 CPU time
1	10	3	0.17	3	0.00	3	0.00
2	5	2	0.05	2	0.00	2	0.00
3	11	3	0.49	5	0.00	5	0.00
4	8	3	0.00	4	0.00	4	0.00
5	6	2	0.00	2	0.06	2	0.00
6	8	3	0.00	3	0.00	3	0.06
7	3	infeasible	0.00	infeasible	0.00	infeasible	0.00
8	3	1	0.05	1	0.00	1	0.00
9	7	2	0.08	3	0.05	3	0.05
10	4	2	0.00	2	0.00	2	0.00
평균	6.5		0.08		0.01		0.01

여 결과를 비교하였다. 각 마이크로셀 당 트래픽은 1), 2) 3), 4)의 경우 각각 평균 9, 12, 9, 7 Erlang의 uniform 분포로부터 발생시켰다.

표는 세가지 환경에서 10가지 문제에 대한 실험 결과와 평균값을 나타낸다. 표에서 핸드오프 되어야 할 마이크로셀의 수는 주어진 섹터와 트래픽을 고려하여 재구성된 섹터를 비교하여 얻어진 수이다. <표 1, 2, 3, 4>에서 동적계획법을 이용한 방법은 마이크로셀 x의 개수가 늘어남에 따라 CPU time이 지수적으로 증가하지만, Algorithm 1, 2는 증가량이 그리 크지 않음을 알 수 있다.

<표 1>에서는 동적계획법과 Algorithm 1, 2에서 제시하는 해가 모두 같고, 계산 시간 또한 아주 작은 값이고 별 차이가 없다. 12 마이크로셀 크기의 시스템에서는 실제로 핸드오프 해야 하는 마이크로셀의 개수가 몇 개 발생하지 않기 때문에 그러한 결과가 나온다고 생각할 수 있다.

<표 2>에서는 10개의 문제 중에서 1개는 실현 불가능해이고, 9개 중에서 3개가 동적계획법과 휴리스틱 알고리즘에서 제시한 해가 다르게 나왔다. 동적계획법은 최적해를 제시하는 반면, 휴리스틱 알고리즘에서는 그 보다 더 많은 핸드오프 단계를 제시하였는데, 그 차이는 1-2 정도이다. 나머지 6개의 문제에서는 같은 해를 제시하고 있다.

<표 3>에서는 2개 문제에서 휴리스틱 알고리즘이 동적계획법 보다 핸드오프 단계를 1단계 더 많이 제시하고 있다. 그러나, 문제 4와 7에서 동적계획법은 핸드오프 해야 하는 마이크로셀의 개수가 많아졌을 때, 계산 시간이 아주 크게 나왔다. 그에 비하여 휴리스틱 알고리즘은 계산시간이 그리 크

〈표 3〉 37-마이크로셀에 대한 동적계획법과 휴리스틱 알고리즘의 실험 결과 비교

Problem	# of microcells to handoff	DP # of steps	DP CPU time	Heuristic 1 # of steps	Heuristic 1 CPU time	Heuristic 2 # of steps	Heuristic 2 CPU time
1	13	3	5.11	4	0.00	4	0.00
2	9	2	0.11	2	0.00	2	0.05
3	12	2	2.14	3	0.00	3	0.00
4	18	3	2225.74	3	0.00	3	0.00
5	13	2	5.33	2	0.00	2	0.00
6	10	2	0.27	2	0.00	2	0.00
7	16	3	169.55	3	0.00	3	0.00
8	9	2	0.11	2	0.00	2	0.00
9	10	2	0.28	2	0.00	2	0.11
10	11	1	1.26	1	0.00	1	0.00
평균	12.1		240.99		0.00		0.02

〈표 4〉 61-마이크로셀에 대한 동적계획법과 휴리스틱 알고리즘의 실험 결과 비교

Problem	# of microcells to handoff	DP # of steps	DP CPU time	Heuristic 1 # of steps	Heuristic 1 CPU time	Heuristic 2 # of steps	Heuristic 2 CPU time
1	17	2	361.63	2	0.06	2	0.06
2	13	2	2.26	2	0.06	2	0.05
3	17	3	45.59	3	0.05	3	0.05
4	16	5	28.89	6	0.11	5	0.11
5	18	3	14.55	4	0.05	4	0.06
6	17	2	397.61	2	0.05	2	0.05
7	18	3	35.76	3	0.05	3	0.06
8	18	3	622.03	3	0.05	3	0.05
9	17	2	815.00	2	0.11	2	0.11
10	16	2	219.32	2	0.06	2	0.05
평균	16.7		254.26		0.07		0.07

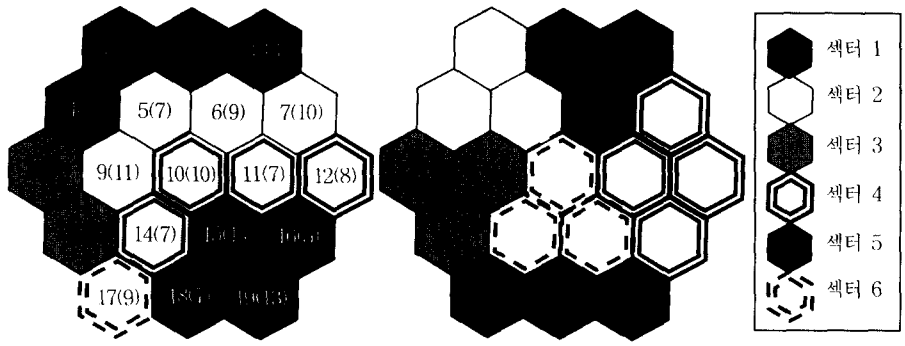
게 증가하지 않았다.

<표 4>는 61개의 마이크로셀이 있는 환경에서 실험을 하였다. 문제 4와 5에서 동적계획법보다 휴리스틱 알고리즘에서 핸드오프 회수가 하나씩 더 크게 나타났다. 그리고, 문제 4에서 휴리스틱 Algorithm 1이 6 step 걸렸지만, Algorithm 2에서는 동적계획법과 같은 5 step만에 문제를 풀었다.

4절에서 계산복잡도를 구했을 때, 계산복잡도는 핸드오프 시켜야 하는 마이크로셀에 의해서 표현되었는데, 실제 동적계획법에서 나타난 계산시간은 마이크로셀의 개수에 정확하게 비례하지는 않게 나타났다. 그러한 이유는 계산복잡도는 계산을 최대한 많이 하게 될 때를 계산했기 때문이다.

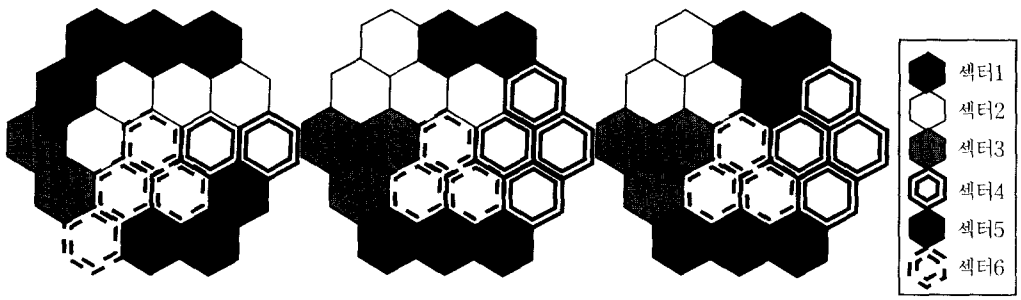
섹터 재구성을 위한 핸드오프의 과정을 19개의 마이크로셀과 6개의 섹터, 2개의 가상기지국을 가진 환경에서의 문제를 예로 설명하면 다음과 같다. [그림 4]의 (a)에 나타나있는 것은 현재의 섹터 구성으로 각 마이크로셀에 셀번호와 트래픽이 괄호 안에 표시되어 있다. 섹터 1과 섹터 2, 섹터 3은 첫 번째 가상기지국에 속하며, 섹터 4와 섹터 5, 섹터 6은 두 번째 가상기지국에 속한다. 하나의 가상기지국에 속한 섹터들은 96개의 채널자원을 공유할 수 있으며, 각각의 섹터는 40개의 채널자원까지 사용할 수 있다. 현재의 섹터 구성으로는 각각의 섹터가 (35, 37, 18, 32, 39, 9)개의 채널자원을 사용하는 편중된 형태를 보여주고 있다. 이러한 상태를 개선시키기 위한 모습으로 [그림 4]의 (b)와 같은 바람직한 섹터 구성이 제시되었다.

[그림 5]에서는 [그림 4]에서 제시된 문제를 세 단계의 핸드오프를 통해서 해결하는 과정을 보여



(a) 현재의 섹터 구성 (b) 요구되는 섹터 구성

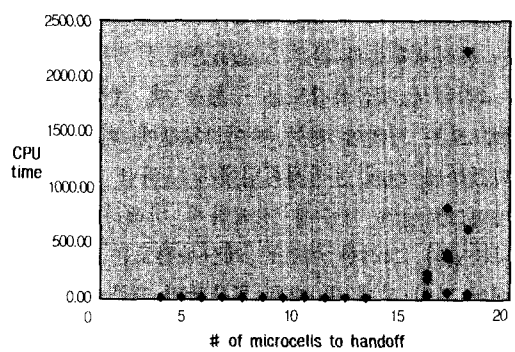
[그림 4] 섹터 재구성의 예



(a) 첫 번째 단계 (b) 두 번째 단계 (c) 세 번째 단계

[그림 5] 핸드오프 과정

주고 있다. 첫번째 단계에서는 상대적으로 여유가 있는 섹터 6으로 핸드오프 해야하는 10번과 14번, 15번 마이크로셀을 핸드오프 시킨다. 그러면 각각의 섹터에서 사용하는 채널자원은 (35, 37, 18, 15, 25, 40)으로 변하게 된다. 섹터 6이 한계치까지 채널자원을 사용하는 대신에 섹터 4와 섹터 5에 여유가 생기게 된다. 두번째 단계에서는 섹터 6에서 섹터 5로 17번 마이크로셀을, 섹터 5에서 섹터 4로 16번 마이크로셀을, 섹터 2에서 섹터 3으로 9번, 섹터 4로 7번 마이크로셀을, 그리고 섹터 1에서 섹터 2로 1번과 4번 마이크로셀을 동시에 핸드오프 시킨다. 이러한 과정에서 섹터나 가상기지국의 용량을 초과하는 상황은 발생하지 않는다. 두번째 단계이 끝나게 되면 각각의 섹터에서 사용하는 채널자원은 (22, 29, 29, 30, 29, 31)로 변하게 된다. 마지막으로 섹터 2에서 섹터 1로 6번 마이크로셀을 핸드오프 시키면 시스템이 요구하는 섹터 구성이 완료된다. 새로운 섹터 구성에서 각각의 섹터가 사용하는 채널자원은 (28, 23, 29, 30, 29, 31)이 되어 현재의 섹터 구성에 비해서 균등부하가 이루어졌다는 것을 알 수 있다.



[그림 6] 동적계획법의 마이크로셀의 개수에 따른 CPU time

[그림 6]은 동적계획법에서 마이크로셀의 개수에 따른 계산 시간을 그래프로 표시하였다. 마이크로셀의 개수가 증가함에 따라 큰 폭으로 증가함을 알 수 있다. 그러나 그러한 증가는 항상 일정하게 비례하는 것이 아니라 문제에 따라 그 증가의 정도가 달라짐을 알 수 있다.

전체적으로 휴리스틱 알고리즘에서 제시하는 해가 동적계획법에서 제시하는 최적해와 그리 많은 차이가 나지 않으면서, 문제의 크기가 커지더라도 휴리스틱 알고리즘에서는 빠른 시간에 해를 제공할 수 있었다.

6. 결 론

광 마이크로 셀룰러 시스템에서는 시스템의 용량을 효율적으로 사용하기 위하여 몇 개의 마이크로셀들을 하나의 섹터로 관리한다. 여기서 섹터간 트래픽의 변화에 따라 한 섹터에 속하는 마이크로셀을 다른 섹터로 옮기게 된다. 이 과정에서 셀단위 핸드오프가 필요하여 그 순서를 결정할 필요가 있다.

이 논문에서는 첫째 셀단위 핸드오프 순서결정 문제를 동적계획법으로 정식화하였다. 둘째로 하나의 마이크로셀을 핸드오프 시켰을 때, 추가로 핸드오프 가능한 마이크로셀들의 트래픽의 합을 이용하여 휴리스틱 알고리즘을 제안하였다. 동적계획법은 최적해를 제시할 수 있지만 계산시간이 문제 크기에 따라 지수적으로 증가하기 때문에 계산시간이 크지 않은 휴리스틱 알고리즘이 효율적인 것으로 나타났다.

논문에서 제시한 휴리스틱 알고리즘은 실험한 40개의 문제 중 6문제를 제외한 모든 경우 동적계획법에서 제시한 해와 같은 해를 제시하였으며 일부 문제에서 근사한 해를 제시하였다. 핸드오프 시켜야 하는 마이크로 셀의 수가 많아질수록 두 방법의 처리시간에 차이가 커지는 것으로 나타났으며, 휴리스틱 방법이 실시간 처리 가능한 것으로 나타났다.

참 고 문 헌

[1] Cheong, J.M., S.H. Seo, J.S. Kim, H.S. Koo, H.S. Park, J.H. Choi, and S. Park, "A Novel CDMA-based Fiber-Optic Microcellular System : FoMiCell^{TM}," *Proc. 49th IEEE VTC*, pp.2200-2203, May 1999.

[2] Cheong, J.M., T.G. Kim, T.H. Park, J.H. Choi, and S. Park, "A Dynamic Resource Management Scheme (DRMS) on Hybrid Fiber Radio (HFR) CDMA Microcellular System," *to appear VTC.*

[3] Denardo, E.V., *Dynamic programming : models and applications*, Prentice-Hall, 1982.

[4] Ohmoto, R., H. Ohtsuka and H. Ichikawa, "Fiber-Optic Microcell Radio Systems with a Spectrum Delivery Scheme," *IEEE Journal on Selected Areas in Communications*, Vol.11, No.7, pp.1108-1117, 1993.

[5] Shanankarannarayanan, N.K., M.R. Philips, T.E. Darcie, and S. Ariyavisitakul, "Multi-port Wireless System Using Fiber/Coaxial Networks for Personal Communication Services and Subscriber Loop Application," *Proc. IEEE GLOBECOM95*, pp.977-981, 1995.

[6] Shirazi, B.A., A.R. Hurson, Krishna M. Kavi, *Scheduling and load balancing in parallel and distributed systems*, Los Alamitos, IEEE Computer Society Press, 1995.

[7] Sniedovich, M., *Dynamic programming*, M. Dekker, 1992.