다차원 색인을 이용한 하향식 계층 클러스터링

(Top-down Hierarchical Clustering using Multidimensional Indexes)

황 재 준 † 문 양 세 †† 황 규 영 †††

(Jae-Joon Hwang) (Yang-Sae Moon) (Kyu-Young Whang)

요 약 최근 공간 데이타 분석, 영상 분석 등과 같은 대용량 데이타를 관리하는 다양한 응용 업무들이 증가함에 따라, 대용량의 데이타베이스를 위한 클러스터링 기법이 많이 연구되고 있다. 그 중에서도 계층 클러스터링 기법은 데이타베이스의 계층 분할을 표현하는 계층 트리를 생성하고 이를 이용하여 효율적인 클러스터링을 수행하는 방법으로서, 지금까지는 주로 트리를 하위 계층으로부터 상위 계층으로 생성해 가는 상향식(bottom-up) 계층 클러스터링 기법들이 연구되었다. 이러한 상향식 클러스터링 방법은 트리를 생성하기 위하여 전체 데이타베이스를 한 번 이상 액세스하여야 할 뿐만 아니라, 하위 계층에서부터 검색을 시작하기 때문에 트리의 많은 부분을 검색하여야 하는 문제점이 있다.

본 논문에서는 대부분의 데이타베이스 응용에서 이미 유지하고 있는 다차원 색인을 이용하여 클러스터링을 수행하는 새로운 하향식(top-down) 계층 클러스터링 기법을 제안한다. 일반적으로 다차원 색인에서는 가까운 객체들이 동일한 (혹은 인접한) 페이지에 저장될 가능성이 큰 클러스터링 성질을 가진다. 이러한 다차원 색인의 클러스터링 성질을 사용하면 각 객체들간의 거리를 일일이 계산하지 않고도 이웃한 객체들을 식별할 수 있다. 우선 객체들의 밀도에 기반하여 클러스터를 정형적으로 정의한다. 이를 위하여, 객체를 포함하는 영역의 밀도를 이용한 영역 대조 분할(*region contrast partition*) 개념을 사용한다. 또, 클러스터링 알고리즘에서의 빠른 검색을 위하여 분기 한정(branch-and-bound) 알고리즘을 사용하며, 여기서의 한계값(bound)을 제안하고 이의 정확성을 이론적으로 증명한다.

실험 결과, 제안한 방법은 상향식 계층 클러스터링 방법인 BIRCH와 비교하여, 정확성 측면에서 우수하거나 유사한 것으로 나타났으며, 데이타 페이지 액세스 횟수를 데이타베이스 크기에 따라 최고 26~187배까지 감소시킨 것으로 나타났다. 이 같은 결과로 볼 때, 제안한 방법은 대용량 데이타베이스에서의 클러스터링 성능을 크게 향상시키는 기법으로서, 일반 데이타베이스 응용에 실용적으로 적용 가능하다고 판단된다.

키워드 : 데이타 마이닝, 클러스터링, 다차원 색인, 밀도 기반 전지

Abstract Due to recent increase in applications requiring huge amount of data such as spatial data analysis and image analysis, clustering on large databases has been actively studied. In a hierarchical clustering method, a tree representing hierarchical decomposition of the database is first created, and then, used for efficient clustering. Existing hierarchical clustering methods mainly adopted the bottom-up approach, which creates a tree from the bottom to the topmost level of the hierarchy. These bottom-up methods require at least one scan over the entire database in order to build the tree and need to search most nodes of the tree since the clustering algorithm starts from the leaf level.

In this paper, we propose a novel top-down hierarchical clustering method that uses multidimensional indexes that are already maintained in most database applications. Generally, multidimensional indexes have the clustering property storing similar objects in the same (or adjacent) data pages. Using this property we can find adjacent objects without calculating distances among

· 본 연구는 첨단정보기술연구센터를 통하여 한국과학재단으로부터 지원 받았음.

† 비 회 원 : 국방과학연구소 책임연구원
jjhwang@mozart.kaist.ac.kr
†† 비 회 원 : (주)인프라밸리 기술연구소 수석연구원
ysmoon@mozart.kaist.ac.kr
††† 종신회원 : 한국과학기술원 전자전산학과 전산학과 교수
kywhang@mozart.kaist.ac.kr

논문접수 : 2002년 2월 14일
심사완료 : 2002년 8월 30일

them. We first formally define the cluster based on the density of objects. For the definition, we propose the concept of the *region contrast partition* based on the density of the region. To speed up the clustering algorithm, we use the branch-and-bound algorithm. We propose the bounds and formally prove their correctness.

Experimental results show that the proposed method is at least as effective in quality of clustering as BIRCH, a bottom-up hierarchical clustering method, while reducing the number of page accesses by up to 26~187 times depending on the size of the database. As a result, we believe that the proposed method significantly improves the clustering performance in large databases and is practically usable in various database applications.

Key words : data mining, clustering, multidimensional index, density-based pruning

1. 서 론

최근 대용량 데이타베이스에서 유용한 패턴을 찾아내기 위한 데이타 마이닝 분야가 각광을 받고 있으며, 클러스터링은 이중 가장 활발히 연구되고 있는 분야 중 하나이다[1]. 비감독학습(unsupervised learning)으로도 알려진 클러스터링은 데이타가 밀집된 부분과 그렇지 않은 부분을 식별하여, 전체 데이타베이스에서의 데이타 분포 패턴을 알아내도록 해준다[2, 3, 4, 5, 6]. 이러한 클러스터링은 사용자의 구매 성향 분석, 의료 데이타 분석, 지리정보 분석, 이미지 분석 등 다양한 응용 분야에서 널리 사용되고 있다. 데이타베이스의 대용량화에 대응하여, 최근의 클러스터링 기법들은 확장성(scalablility)의 지원에 연구의 초점을 두고 있다[7, 8]. 과거의 클러스터링 기법들은 대개 정확한 클러스터링을 수행하는데 그 목적이 있었다. 그러나, 데이타베이스가 대용량화되면서 정확성을 우선으로 한 이들 알고리즘은 많은 수행시간으로 인하여 실용적으로는 더 이상 사용하기 어렵게 되었다. 이에 따라, 대용량 데이타베이스를 분석하는데 있어서, 정확성을 크게 해치지 않으면서도 빠른 클러스터링 방법이 요구되고 있다.

계층 클러스터링 기법은 데이타베이스의 계층 분할을 표현하는 계층 트리를 생성하고 이를 이용하여 효율적인 클러스터링을 수행하는 방법으로서, 데이타베이스에 대한 클러스터링 성능 향상을 위하여 활발히 연구되어 왔다[3, 6, 8, 9, 10]. 본 논문은 이러한 계층 클러스터링 기법의 연구에 초점을 맞추며, 분할 클러스터링 기법 등 그 이외의 클러스터링 기법들에 대해서는 참고문헌 [2, 4, 5, 11, 12, 13]을 참조한다. 계층 클러스터링 기법은 크게 상향식 계층 클러스터링 기법과 하향식 계층 클러스터링 기법으로 나눌 수 있다[14]. 우선, 상향식 계층 클러스터링 기법은 데이타베이스의 각 객체를 트리의 최하위 노드에 대응하는 별도의 클러스터라 간주하고 시작하여, 주어진 척도(예를 들면, 두 객체간의 거리)에 따라 각 노드에 저장된 객체 또는 객체들의 집합을 반복적으로 병합해 가며 계층 트리를 구성한다. 이러한 과정은 모든 객체들이 트리의 최상위 노드에 대응하는 하나의 클러스터로 병합되거나, 또는 주어진 종료 조건(예를 들면, k개의 클러스터)이 만족될 때까지 수행된다. 다음으로, 하향식 계층 클러스터링 기법은 데이타베이스의 모든 객체들이 트리의 최상위 노드에 대응하는 하나의 클러스터라 간주하고 시작하여, 주어진 척도에 따라 각 노드에 저장된 객체들의 집합을 더 작은 클러스터들로 반복적으로 분할해 가며 계층 트리를 구성한다. 이러한 과정은 각 객체가 트리의 최하위 노드에 대응하는 별도의 클러스터가 되거나, 또는 주어진 종료 조건이 만족될 때까지 수행된다. 대용량 데이타베이스인 경우 모든 객체를 대상으로 계층 트리를 구성하게 되면 큰 저장 공간이 요구되고 노드의 분할 또는 병합에 따른 성능 저하의 문제점이 있다.

대용량 데이타베이스의 빠른 클러스터링을 위해서는 데이타베이스 액세스 횟수와 트리 검색 과정에서의 검색 공간을 줄이는 것이 필요하다. 일반적으로, 데이타 웨어하우스, 지리 정보 시스템 등 대부분의 데이타베이스 응용에서는 대용량의 데이타베이스 질의 처리를 위하여 다차원 색인을 관리한다. 이러한 다차원 색인에서는 거리가 가까운 객체들은 동일한 단말 페이지(leaf page) 혹은 내부 페이지(internal page)에 저장될 가능성이 크며, 이를 다차원 색인의 클러스터링 성질(*clustering property*)이라 부른다[11, 15]. 그리고, 다차원 색인의 클러스터링 성질을 사용하면 내부 페이지 혹은 단말 페이지의 정보를 활용하여 이웃한 객체들을 식별할 수 있으며, 이에 기반하여 전체 데이타베이스의 액세스와 많은 거리 계산 없이 빠른 클러스터링을 수행할 수 있다.

본 논문에서는 다차원 색인이 갖는 클러스터링 성

절을 이용하여 데이타베이스의 페이지 액세스 횟수를 크게 줄이는 하향식 계층 클러스터링 기법을 제안한다. 먼저, 데이타베이스 객체들의 밀도(density)에 기반하여 클러스터를 정형적으로 정의하였다. 이를 위하여, 객체를 포함하는 색인 영역의 밀도에 의거하여 영역 대조 분할(*region contrast partition*)이라는 개념을 제시하였다. 다음으로, 정의한 클러스터를 빠르게 찾는 하향식 클러스터링 방법을 제안하였다. 제안하는 방법은 밀도 기반 전지를 이용한 분기 한정(branch-and-bound) 알고리즘을 사용하며, 본 논문에서는 알고리즘에서의 한계값(bound)을 제시하고 이 한계값의 정확성을 이론적으로 증명하였다. 마지막으로, 실험을 통하여 제안한 방법이 정확하며 기존 방법에 비해 매우 우수한 성능을 보임을 입증하였다. 특히, 대용량 데이타 집합일수록 페이지 액세스 횟수의 감소 비율이 높아져 성능 향상 효과가 크게 나타남을 보였다.

본 논문의 구성은 다음과 같다. 제2장에서는 대용량 데이타베이스에 대한 기존의 계층 클러스터링 연구들을 살펴보고, 제3장에서는 본 논문에서 제안하는 다차원 색인을 이용한 하향식 클러스터링 기법을 기술한다. 제4장에서는 제안한 방법과 기존의 연구 결과를 비교한 성능 평가 결과를 제시하고, 마지막으로 제5장에서는 결론을 기술한다.

2. 관련 연구

초기의 대표적인 계층 클러스터링 방법으로는 통계 패키지인 S-PLUS에 구현된 AGNES와 DIANA가 있다[14]. AGNES는 각 객체를 별도의 클러스터라 간주하고 시작하여, 모든 객체들이 하나의 클러스터에 포함될 때까지 이들 클러스터를 반복적으로 병합해 가는 상향식 계층 클러스터링 기법이다. 반면에 DIANA는 AGNES의 역순으로 클러스터링을 수행하는 하향식 기법이다. 그러나, 이들 두 알고리즘은 병합(merge) 또는 분할(split) 수행시 많은 객체들간의 거리 계산을 요구하기 때문에 대용량 데이타베이스에 대한 확장성이 없으며, 객체들간의 거리 척도만을 사용한 단순한 병합 또는 분할 전략으로 인하여 찾은 클러스터의 정확성이 저하될 가능성이 높다는 단점이 있다.

최근에 연구된 대용량 데이타베이스에 대한 효율적인 계층 클러스터링 방법으로 샘플링을 사용하는 방법들[3, 7, 16]과 클러스터 요약 정보를 정의하여 사용하는 방법들[6, 8]이 있다. 샘플링을 사용하는 방법은 대용량의 데이타베이스로부터 샘플을 추출하고 추출된 샘플을 대상으로 계층 트리를 구성하여 클러스터링을 수행하는 기법이다. 이 방법은 매우 간단하고 쉽게 적용할 수 있다는 장점이 있는 반면에, 찾은 클러스터의 정확도가 샘플링의 정확도에 크게 의존한다는 단점을 갖고 있다. 요약 정보를 사용하는 방법은 찾고자 하는 클러스터 형태를 잘 표현할 수 있는 요약 정보를 정의하고, 이를 대상으로 계층 트리를 구성하여 클러스터링을 수행하는 기법이다. 이러한 기법들은 요약 정보의 특성에 따라 클러스터 형태가 결정된다는 단점이 있다.

BIRCH[6]는 대용량 데이타베이스에 대한 효율적인 클러스터링을 위하여 요약 정보를 사용하는 대표적인 상향식 계층 알고리즘이다. BIRCH는 원 데이타베이스가 갖는 클러스터 요약 정보를 계산하여 이를 노드 값으로 갖는 계층 데이타 구조를 생성한 후, 원 데이타베이스 대신 이 구조를 이용하여 클러스터링을 수행한다. *CF*-트리라 불리는 계층 데이타 구조는 여러 객체들을 대표하는 클러스터 요약 정보인 *C*lustering *F*eature(*CF*)들을 저장하고 있다. 하나의 *CF*는 여러 개의 객체를 대표하므로, 하나의 *CF*가 대표하는 객체의 개수를 많게 하면 고려해야 할 *CF*의 개수를 줄일 수 있다. 이 알고리즘은 원 데이타베이스의 객체 수보다 상대적으로 훨씬 적은 수의 *CF*들을 대상으로 클러스터링을 수행하므로 빠를 뿐만 아니라, *CF*-트리의 크기를 가용 메모리에 맞추어 조정할 수 있다. 따라서, 대용량 데이타베이스에 대해서도 빠른 클러스터링이 수행될 수 있도록 보장한다. 또한, BIRCH는 노이즈 객체들을 처리할 수 있는 최초의 알고리즘이다[13]. 그러나, BIRCH는 초기 *CF*-트리를 구성하기 위하여 전체 데이타베이스를 최소 한 번 스캔하여야 하며, 구형(spherical)이 아닌 클러스터인 경우에는 정확한 결과를 제공하지 못한다는 단점이 있다[3, 13]. 참고문헌 [8]에서는 BIRCH의 *CF* 개념을 일반화시켜 임의의 거리 공간(arbitrary metric spaces)에서의 대용량 데이타베이스에 대한 확장성 있는 클러스터링 방법을 제시하였다. CURE[3]는 임의 모양의 클러스터를 찾을 수 있는 계층 알고리즘이다. 이 알고리즘은 하나의 *CF*값으로 클러스터를 표현하는 BIRCH와 달리, 여러 개의 대표 객체들을 선정하여 클러스터를 표현하고 계층 트리를 구성할 때 이들을 사용하여 클러스터간의 거리를 계산함으로써 다양한 크기와 임의 모양의 클러스터를 찾도록 한다. 또, 대용량 데이타베이스에 대한 확장성을 제공하기 위하여 클러스터링 초기 단계에서 임의 샘플링을 통해 얻은 샘플 공간을 분할하여 각 분할을 대상으로 부분 클러스터를 찾는 사전

클러스터링(pre-clustering)을 수행한다. CURE는 또한 노이즈 객체들이 클러스터에 포함되지 않도록 여과하는 기능을 제공한다. 그러나, CURE는 클러스터링 결과가 샘플링의 정확도에 의존하며, 이에 따라 충분한 크기의 샘플을 사용하여야 한다는 제약을 갖는다.

샘플링 또는 클러스터 요약 정보를 사용하는 방법에서의 문제점은 샘플링 비율이나 요약 정보 생성시의 데이타 압축 비율을 높게 할수록 클러스터링 결과가 저하된다는 것이다. 최근 참고문헌 [7]은 임의 샘플링 또는 BIRCH의 *CF*를 이용하여 클러스터링 정확도를 저하시키지 않으면서 데이타 압축 비율을 높임으로써 대용량 데이타베이스에 대한 클러스터링 성능을 향상시키는 방법을 제시하였다. 이 외에 참고문헌 [16]에서는 샘플링 사용시 균일 샘플링(uniform sampling)으로 인하여 상대적으로 크기가 작거나 밀도가 낮은 클러스터들을 찾지 못하게 되는 문제점을 해결하기 위해 밀도 편향 샘플링(density biased sampling) 방법을 제시하였다.

BIRCH나 CURE와 같은 이제까지의 계층 클러스터링 기법들은 모두 상향식 접근 방법을 취하고 있으며, 데이타베이스 객체들이 어떻게 클러스터를 형성하는지를 알기 위해서는 데이타베이스 전체를 한 번 이상 액세스 하여야만 한다. 또한, 트리의 하위 계층에서부터 검색을 시작하기 때문에 트리의 많은 부분을 검색하여야 하며, 이로 인하여 주기억장치상에 트리를 생성하고 유지해야 하는 단점이 있다.

3. 다차원 색인을 이용한 하향식 클러스터링 기법

본 장에서는 기존 상향식 계층 클러스터링 기법의 단점을 해결하는 다차원 색인 기반 하향식 계층 클러스터링 기법을 제안한다. 제 3.1절에서는 사용하는 용어들을 정의하고, 제안하는 클러스터링 기법의 근거를 이루는 영역 대조 분할과 이에 기반한 클러스터를 정의한다. 제 3.2절에서는 정의에 따른 클러스터를 찾아내는 단말 대조 알고리즘을 기술한다. 제 3.3절에서는 효율적인 클러스터링을 위해 도입하는 밀도 기반 전지 개념에 대해 설명하고, 마지막으로 제 3.4절에서는 이를 이용한 밀도 기반 전지 알고리즘을 제안한다.

3.1 문제 정의

3.1.1 용어 정의

그림 1은 다차원 파일 구조의 일반적인 형태와 각 구성 요소의 명칭을 나타낸다. 다차원 파일 구조는 일반적으로 다차원 색인을 수행하는 색인 페이지(*index page*)와 실제 데이타 객체를 저장하는 데이타 페이지(*data page*)의 두 부분으로 크게 나눌 수 있다. 색인 페이지는 각 페이지가 속하는 계층에 따라 루트, 내부, 그리고 단말 페이지로 구분된다. 우선, 루트 페이지(*root page*)는 가장 상위 계층에 존재하는 색인 페이지를 말하며, 이 페이지의 내부는 (키, 자식 포인터)의 구조를 갖는 루트 엔트리(*root entry*)들로 구성되어 있다. 다음으로, 단말 페이지(*leaf page*)는 가장 하위 계층을 구성하는 색인 페이지를 말하며, 각 페이지의 내부는 (키, 객체 식별자)의 구조를 갖는 단말 엔트리(*leaf entry*)들로 구성되어 있다. 내부 페이지(*internal page*)는 루트 페이지와 단말 페이지 사이의 중간 계층에 존재하는 색인 페이지를 말하며, 각 페이지는 (키, 자식 포인터)의 구조를 갖는 내부 엔트리(*internal entry*)들로 구성되어 있다. 일반적으로 루트 페이지는 내부 페이지에 포함시키기도 하나, 설명의 편의상 이를 구분하여 사용하기로 한다.

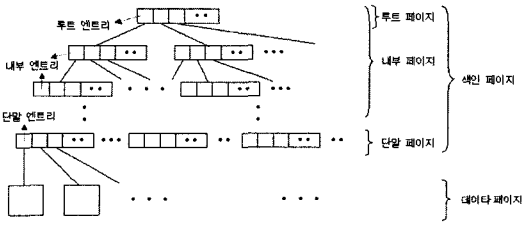


그림 1 다차원 파일 구조의 구성.

색인 페이지의 특정 엔트리가 표현하는 영역을 색인 영역(*index region*) 혹은 간단히 영역이라 한다. 그리고, 단말 엔트리가 나타내는 영역을 단말 영역(*leaf region*), 내부 엔트리가 나타내는 영역을 내부 영역(*internal region*)이라 정의한다. 다음으로, 영역 R의 밀도(*density of region R*)란 영역 R에 포함된 객체수와 영역 크기와의 비율로 정의한다. 마지막으로, 대부분의 다차원 색인은 영역의 형태가 초사각형(hyper-rectangle)이므로[17, 18, 19], 여기서도 모든 영역의 형태가 초사각형이라 가정한다.

본 논문에서는 영역 기준 분할전략[15]을 취하는 색인 구조를 사용하여 클러스터링을 수행한다. 영역 기준 분할전략은 영역의 분할시 영역의 크기를 반분하는 위치를 분할 경계로 한다. 그리고, k-차원 색인을 가정할 경우, 임의의 두 영역이 서로 포함 관계에 있을 때를 제외하고는 k-차원 상에서 서로 겹치는 경우는 발생하지 않는 특징을 가진다.

다차원 색인에서의 두 영역간의 인접을 다음과 같이 정의한다.

정의 1 k-차원의 두 영역을 $R_A=\{(a_{x_1}, a_{y_1}), ..., (a_{x_k}, a_{y_k})\}$, $R_B=\{(b_{x_1}, b_{y_1}), ..., (b_{x_k}, b_{y_k})\}$라 할 때, 두 영역의 단 1개 차원만이 조건 1을 만족하고, 나머지 $(k-1)$개 차원은 조건 2를 만족하면, 두 영역 R_A와 R_B는 서로 인접(*adjacent*)한다고 정의하고, 이때 이들 두 영역을 서로의 인접 영역(*adjacent region*)이라 정의한다. 그리고, 이를 $R_A \oplus R_B$라 표기한다.

조건 1 $(a_{x_i} = b_{y_i}) \vee (b_{x_i} = a_{y_i}), 1 \le i \le k$

조건 2 $(a_{x_j} \le b_{x_i} < a_{y_i}) \vee (b_{x_j} \le a_{x_i} < b_{y_i}), 1 \le j \le k, j \ne i$

다음으로, 두 영역간의 이행적 인접을 다음과 같이 정의한다.

정의 2 임의의 두 영역을 R_A와 R_B라 할 때, $R_A \oplus R_B$이거나, $R_A \oplus R_1$, $R_1 \oplus R_2$, ..., $R_k \oplus R_B$와 같은 인접 영역들의 순열 $\{R_A, R_1, R_2, ..., R_k, R_B\}$가 적어도 하나 존재한다면, R_A와 R_B는 서로 이행적 인접(*transitively adjacent*)한다고 정의하고, 이를 $R_A \otimes R_B$라 표기한다.

예제 1: 그림 2는 영역 기준 분할전략을 사용하는 2-차원 색인 구조를 나타낸다. 그림에서 영역 A와 B는 인접한다($A \oplus B$). 그 이유는 A와 B의 두 차원이 정의 1의 조건을 만족하기 때문이다. 즉, A와 B의 첫 번째 차원은 정의 1의 조건 1을 만족하며, 두번째 차원은 정의 1의 조건 2를 만족하기 때문이다. 또한, 영역 B와 C, C와 D도 정의에 따라 인접하는 영역으로, 각각 $B \oplus C$와 $C \oplus D$로 나타낼 수 있다. 다음으로, 영역 B와 D, A와 C, A와 D 등은 정의 1의 조건을 만족하지 않으므로 인접하지는 않으나, 이들 각각은 서로 이행적 인접한다. 예를 들어, 영역 A와 D의 경우 $A \oplus B$, $B \oplus C$, $C \oplus D$의 인접 영역 관계가 있는 순열 $\{A, B, C, D\}$가 존재하므로 A와 D는 이행적 인접한다. 반면에, 영역 A와 E간에는 그러한 순열이 존재하지 않으므로 이행적 인접하지 않는다. □

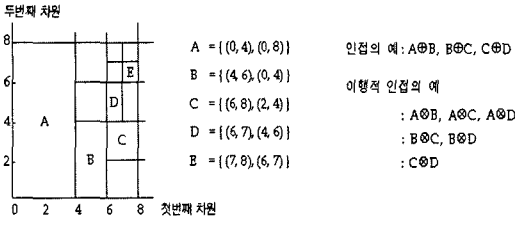


그림 2 2차원 색인 영역간의 인접과 이행적 인접의 예

마지막으로, 전체 색인 영역을 객체가 밀집된 영역과 그렇지 않은 영역으로 분할하는 기준으로서, 다음과 같은 클러스터링 인수를 정의한다.

정의 3 객체가 밀집된 영역으로 판별된 영역들에 포함된 총 객체수의 하한값과 데이타베이스에 저장된 총 객체수와의 비율을 클러스터링 인수(*clustering factor*)라 정의하고, 이를 ρ라 표기한다.

클러스터링 인수는 데이타베이스에 저장된 총 객체수 중에서 객체가 밀집된 영역들의 집합에 포함되어야 할 최소한의 객체수를 보장해 준다. 뿐만 아니라, 객체가 밀집되어 있지 않은 영역들을 판별하여 이들 영역이 클러스터에 포함되지 않도록 함으로써, 일정 비율의 노이즈(noise) 객체가 존재하는 것을 허용하며, 이들 객체들을 제거 가능한 객체(removable objects)라 부르고, 그 개수를 제거 가능한 객체수(number of removable objects) NRO로 표기한다. 클러스터링 인수 및 제거 가능한 객체수의 활용에 대해서는 제3.1.2절에서 자세히 설명한다.

본 논문에서 사용하는 주요 표기법을 정리하면 표 1과 같다.

표 1 주요 표기법

기호	정의/의미
N	데이타베이스에 저장된 총 객체수
k	다차원 색인(클러스터)의 차원
$s(R)$	영역 R의 크기
$n(R)$	영역 R에 포함된 객체수
$d(R)$	영역 R의 밀도 (= $n(R)$ / $s(R)$)
$R_i \oplus R_j$	두 영역 R_i와 R_j가 인접함을 의미
$R_i \otimes R_j$	두 영역 R_i와 R_j가 이행적 인접함을 의미
ρ	사용자에 의해 주어지는 클러스터링 인수 ($0 \le \rho \le 1$)
NRO	제거 가능한 객체수

3.1.2 영역 대조 분할과 클러스터의 정의

본 절에서는 영역들의 밀도 정보와 클러스터링 인수를 사용하여 영역들을 분할하는 방법을 제안하고, 이 분할에 기반하여 클러스터를 정의한다.

정의 4 전체 색인 영역 $\boldsymbol{R}$이 서로 소인 영역들의 집합 $\{R_1, R_2, ..., R_m\}$으로 이루어져 있고 제거 가능한 객체수를 NRO라 했을 때, $\boldsymbol{R}$의 분할인 집합 $\{\boldsymbol{R_D}, \boldsymbol{R_S}\}$가 다음 세 가지 조건을 만족할 경우 이를 영역 대조 분할(*region contrast partition*)이라 정의한다.

조건 1 $\forall i, j \; (\; (R_i \in \boldsymbol{R_S}) \wedge (R_j \in \boldsymbol{R_D}) \Rightarrow d(R_i) \le d(R_j) \;)$

조건 2 $\sum_{R_i \in R_S} n(R_i) \leq NRO$

조건 3 $\sum_{R_i \in R_i} n(R_i) + n(R_p) \leq NRO$, R_p는 R_D에서의 최저 밀도 영역

영역 대조 분할의 결과로 얻은 두 집합 R_D와 R_S에 속한 영역들을 각각 밀집 영역(*dense region*)과 희소 영역(*sparse region*)이라 정의한다. 특히, 집합 R이 단말 영역들의 집합일 경우에는, R_D에 속한 영역들을 밀집 단말 영역(*dense leaf region*), R_S에 속한 영역들을 희소 단말 영역(*sparse leaf region*) 이라 정의한다. 또, 정의 4의 조건 3에서 R_p를 집합 R의 분할 경계 영역(*partition boundary region*)이라 한다.

정리 1 영역 기준 분할전략을 사용하는 다차원 색인은 항상 영역 대조 분할을 구하는 것이 가능하다.

증명: 부록 참조. □

이제 영역 대조 분할의 결과로 얻은 밀집 영역들의 집합 R_D를 사용하여 본 논문에서 제안하는 클러스터와 클러스터링을 정의한다.

정의 5 밀집 단말 영역들의 집합을 $R_D = \{R_1, R_2, ..., R_p\}$라 할 때, 다음 세 가지 조건을 만족하는 영역들의 집합 C를 클러스터(*cluster*)라 정의한다.

조건 1 $C \subseteq R_D$

조건 2 $(R_i \in C) \wedge (R_i \otimes R_j) \Rightarrow R_j \in C$

조건 3 $\sum_{R_i \in C} n(R_i) > \varepsilon, (\varepsilon \ll N)$

정의 5의 조건 2는 이행적 인접한 모든 밀집 단말 영역들은 하나의 클러스터로 형성됨을 나타내는 것으로서, 이를 클러스터의 최대화 조건(*maximality condition*)이라 정의한다. 또한, 조건 3은 하나의 클러스터가 일정 개수(ε) 이상의 객체를 포함하여야 함을 의미하는 것으로서, 이 ε값을 클러스터 형성자(*cluster qualifier*)라 부른다. 조건 1과 조건 2를 만족하나 조건 3을 만족하지 않는 영역들의 집합을 밀집 조각(*dense chip*)이라 정의한다.

정의 6 클러스터링(*clustering*)이란 단말 영역들의 집합을 영역 대조 분할하여 밀집 단말 영역들의 집합 R_D를 구하고, R_D에 포함된 모든 클러스터들을 찾아내는 작업이다.

3.2 단말 대조 알고리즘

본 절에서는 제3.1절에서 정의한 클러스터링을 수행하는 직관적 알고리즘인 단말 대조 알고리즘(*leaf contrast algorithm*)을 제안한다. 단말 대조 알고리즘은 색인의 모든 단말 영역들을 대상으로 정의 4의 영역 대조 분할을 수행하여 밀집 단말 영역들의 집합을 구한 후, 정의 5를 따르는 클러스터들을 찾는 알고리즘이다.

```
Algorithm LeafContrast
Input:   md_index  : multidimensional index to be used for clustering
         N         : the number of objects stored in the database
         ρ         : clustering factor
Output:  Set of Clusters
begin
1:  NRO := (1-ρ)*N;
2:  Construct a list L of leaf regions by reading all leaf pages from md_index;
3:  Make a sorted list L_new(={R_1, R_2, ..., R_m}) by sorting L in the ascending order of region density;
4:  p := Find_Partition_Region(L_new, NRO);
5:  R_dense := {R_1, R_2, ..., R_p};
6:  C := Find_Clusters(R_dense);
7:  Return C;
end

Algorithm Find_Partition_Region
Input:   L         : List of regions
         NRO       : the number of removable objects
Output:  Partition Boundary
begin
1:  no_objs := 0;                    p := 0;
2:  while (no_objs ≤ NRO) begin
3:      p := p + 1;
4:      no_objs := no_objs + n(R_p);
5:  end
6:  Return p;
end

Algorithm Find_Clusters
Input:   R_dense   : set of dense regions
         ε         : cluster qualifier
Output:  Set of clusters
begin
1:  j := 0;
2:  while (R_dense is not empty) begin
3:      j := j + 1;
4:      C_j := {a region R_i of R_dense};
5:      Find all the transitively adjacent regions of R_i;
6:      Insert the regions into C_j and remove the regions from R_dense;
7:  end
8:  Remove the dense chips;
9:  Return C;
end
```

그림 3 단말 대조 알고리즘

제안하는 단말 대조 알고리즘 *LeafContrast*의 수행 과정과 관련 함수들은 그림 3과 같다. 먼저 클러스터링 인수 ρ와 데이타베이스에 저장된 총 객체수 N을 사용하여 제거 가능한 객체수 NRO를 계산한다(스텝 1). 그리고, 다차원 색인 *md_index*의 모든 단말 페이지를 액세스하여 단말 영역들의 리스트를 구성하고, 이를 영역 밀도를 사용하여 오름차순으로 정렬한다(스텝 2~3). 다음으로, 정렬된 단말 영역들의 리스트를 사용하여 분할 경계 영역 R_p를 찾아낸다(스텝 4). 마지막으로, R_p보다 높은 밀도를 갖는 단말 영역들의 집합인 *R_dense*를 구하고, 이들 영역들을 대상으로 클러스터링을 수행한다(스텝 5~6). 함수 *Find_Partition_Region*은 주어진 NRO를 이용하여 정렬된 영역들의 리스트로부터 분할 경계 영역을 찾아낸다. 다음으로, 함수 *Find_Clusters*는 밀집 영역들의 집합 *R_dense*로부터 최대화 조건을 만족하기 위해 이행적으로 인접하는 모든 영역들을 찾은 후, 밀집 조각들을 제거하여 클러스터를 얻는다.

예제 2: 그림 4는 단말 대조 알고리즘이 수행되는 순서를 예시한 것이다. 그림 4(a)는 데이타 공간을 분할하고 있는 다차원 색인의 단말 영역들을 나타내며, 그림 4(b)는 알고리즘에서의 영역 대조 분할 수행 결과로, 밀집 단말 영역들과 희소 단말 영역들이 구분되었음을 나타낸다. 그림 4(c)는 밀집 단말 영역들로부

터 이행적으로 인접한 모든 영역들을 찾아내어 최종 결과로 얻은 클러스터들을 나타낸다. □

정리 2 단말 대조 알고리즘으로 구한 클러스터들은 정의 5의 클러스터들과 동일하다.

증명: 정의 1~정의 6을 그대로 구현한 알고리즘이므로, 자세한 증명은 생략한다. □

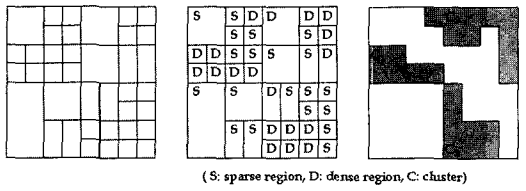


그림 4 단말 대조 알고리즘 수행 과정.

3.3 밀도 기반 전지 개념

본 절에서는 단말 대조 알고리즘의 성능을 개선하기 위한 밀도 기반 전지 개념을 기술한다. 단말 대조 알고리즘에서는 클러스터링을 위하여 다차원 색인 전체(혹은 모든 단말 페이지)를 액세스해야만 한다. 대신 색인의 모든 단말 엔트리를 액세스하지 않고 내부 기준에 의하여 자신의 단말 영역들이 모두 밀집 단말 영역이거나 모두 희소 단말 영역임을 판단하여 검색을 생략하는 것을 밀도 기반 전지(*density-based pruning*)라 정의한다. 그리고, 어떠한 내부 영역에 속한 단말 영역들이 모두 밀집 단말 영역들인 경우에 이 내부 영역을 밀집 내부 영역(*dense internal region*)이라 정의하고, 반대로 모두 희소 단말 영역들인 경우에는 이 내부 영역을 희소 내부 영역(*sparse internal region*)이라 정의한다.

예제 3: 그림 5는 단말 영역을 나타내는 단말 페이지와 이를 포함하는 상위 수준의 내부 페이지를 나타낸 것으로, D는 *dense*를 S는 *sparse*를 의미한다. 그림에서 보는 바와 같이 내부 영역 D_0에 속한 단말 영역들인 D_1, D_2, D_3, D_4는 모두 밀집 단말 영역들이고, S_0에 속한 S_1, S_2, S_3, S_4는 모두 희소 단말 영역들이다. 따라서, D_0는 밀집 내부 영역이며, S_0는 희소 내부 영역이다. □

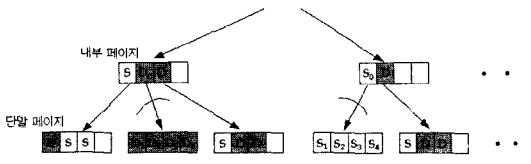


그림 5 밀집 내부 영역과 희소 내부 영역

3.4 밀도 기반 전지 알고리즘

본 절에서는 정의한 밀도 기반 전지를 분기 한정 알고리즘에 적용하기 위한 한계값(bound)을 제시하고, 이를 이용하여 보다 효율적인 클러스터링을 수행할 수 있는 밀도 기반 전지 알고리즘을 제안한다. 제안하는 밀도 기반 전지 알고리즘(*density-based pruning algorithm*)은 밀도 기반 전지를 이용하되 단말 대조 알고리즘과 동일한 클러스터를 구할 수 있는 방법이다.

밀도 기반 전지 알고리즘에서는 전지의 가능 여부를 판단하기 위하여 다차원 색인의 내부 엔트리에서 최고 밀도(*d_highest*)와 최저 밀도(*d_lowest*)의 두 가지 밀도 정보를 관리한다. 이들 두 값은 각각 자신의 영역에 포함된 단말 영역들 중에서 최고 밀도를 갖는 영역과 최저 밀도를 갖는 영역의 밀도를 나타낸다. 밀도 기반 전지 알고리즘은 하향식 넓이 우선 탐색(breadth first search) 방식으로 색인을 검색하여 영역 대조 분할을 수행한 후, 분할 경계 영역의 밀도를 한계값으로 취하여 하위 계층의 전지 여부를 결정하는 분기 한정 알고리즘이다. 이렇게 함으로써 하향식 깊이 우선 탐색 방식으로 모든 단말 영역들을 검색하지 않고도 정의 5의 클러스터를 찾을 수 있다.

제안하는 밀도 기반 전지 알고리즘 *DB_Pruning*의 수행 과정은 그림 6과 같다. 먼저 클러스터링 인수 ρ와 데이타베이스에 저장된 총 객체수 N을 사용하여 제거 가능한 객체수 NRO를 계산하고, 색인의 최상위 계층인 루트 계층을 현재 계층으로 설정한다(스텝 1~3). 그리고, 다차원 색인 *md_index*로부터 현재 계층의 내부 페이지를 모두 액세스하여 내부 영역들의 리스트 L을 구성한다(스텝 5). 다음으로, L을 *d_highest* 및 *d_lowest* 각각에 대해 오름차순으로 정렬한 리스트 L_{high}와 L_{low}를 만든 후, 각 리스트에서 분할 경계 영역 R_p를 찾아 이 영역의 밀도를 각각 B_{high} 및 B_{low}로 한다(스텝 6~11). 여기서 B_{high}는 상위 한계값(upper bound)을 의미하고, B_{low}는 하위 한계값(lower bound)을 의미하는 것으로서 다음 스텝에서의 전지 여부를 판단하기 위하여 사용된다. 밀도 기반 전지는 다음과 같이 두 단계로 나뉘어 수행한다. 첫째, L_{low}의 엔트리들 중에서 $d_lowest > B_{high}$인 영역은 밀집 내부 영역을 나타내므로, 해당 영역을 밀집 영역들의 집합으로 삽입하고 전지한다(스텝 12~15). 둘째, L_{high}의 엔트리들 중에서 $d_highest < B_{low}$인 영역은 희소 내부 영역을 나타내므로, 해당 영역에 포함된 객체수만큼 NRO

를 감소시킨 후 전지한다(스텝 16~19). 현재 계층을 검색한 후에도 제거 가능한 객체수만큼 모두 제거되지 않았다면(즉, $NRO > 0$), 다음 하위 계층을 현재 계층으로 설정하고(스텝 20) 위의 스텝 5부터 19까지의 과정을 반복한다. 이와 같은 반복 과정은 현재 계층이 단말 영역에 도달할 때까지 계속 수행된다. 현재 계층이 단말 영역에 도달하여 반복 과정이 종료한 후에도 $NRO > 0$이라면, 남아 있는 모든 단말 영역들의 리스트를 구성한 후 영역 대조 분할을 수행하여 분할 경계 영역 R_p를 찾는다(스텝 23~25). 마지막으로, R_p보다 높은 밀도를 갖는 단말 영역들의 집합을 *R_dense*에 추가하고, 이들 영역들을 대상으로 클러스터링을 수행한다(스텝 26~29). 밀도 기반 전지 알고리즘에서 사용되는 함수 *Find_Partition_Region*과 *Find_Clusters*는 단말 대조 알고리즘에서와 동일한 함수이다.

정리 3 밀도 기반 전지 알고리즘에서 자신의 *d_lowest*값이 B_{high}보다 큰 영역은 밀집 내부 영역이고, 자신의 *d_highest*값이 B_{low}보다 작은 영역은 희소 내부 영역이다.

증명: 먼저 *d_lowest*값이 B_{high}보다 큰 영역이 밀집 내부 영역임을 증명한다. 리스트 L_{high}에서 B_{high}이하의 밀도를 갖는 영역들의 집합을 $\boldsymbol{R}^I = \{ R_1^I, ..., R_x^I \}$라 하고, $\boldsymbol{R}^I$에 속한 모든 단말 영역들의 집합을 $\boldsymbol{R}^L = \{ R_1^L, ..., R_y^L \}$라 하자. $\boldsymbol{R}_S$를 희소 단말 영역들의 집합이라 하고 $\boldsymbol{R}_D$를 밀집 단말 영역들의 집합이라 하면, B_{high}의 정의에 의해 $n(\boldsymbol{R}^I) = n(\boldsymbol{R}^L) > NRO$이므로, $\boldsymbol{R}^L$에는 $R_i^I \in \boldsymbol{R}^L - \boldsymbol{R}_S$인 $R_i^L (\in \boldsymbol{R}_D)$가 반드시 존재한다. 그리고, 리스트 L_{low}에서 *d_lowest* > B_{high}인 임의의 영역을 S^I라 하고, S^I에 속한 모든 단말 영역들의 집합을 $\boldsymbol{S}^I = \{ S_1^L, ..., S_z^L \}$라 하자. 이때, $\boldsymbol{S}^I$의 임의의 단말 영역 S_j^L이 희소 단말 영역($S_j^L \in \boldsymbol{R}_S$)이라 가정하자. 그러면, $S_j^L \in \boldsymbol{R}_S$이고 $R_i^L \in \boldsymbol{R}_D$이므로, $d(R_i^L) > d(S_j^L)$이 성립한다. 그런데, B_{high}와 *d_lowest*의 정의에 의해, 집합 $\boldsymbol{R}^L$의 영역 R_i^L에 대해 $d(R_i^L) \le B_{high}$이 성립하고, 집합 $\boldsymbol{S}^I$의 영역 S_j^L에 대해 $d(S_j^L) \ge d_lowest$이 성립한다. 그리고, 영역 S^I에 대해 $d_lowest > B_{high}$이므로, $d(R_i^L) < d(S_j^L)$가 성립한다. 그러나, 이는 S_j^L가 희소 단말 영역이라는 가정 ($d(R_i^L) > d(S_j^L)$)에 위배된다. 따라서, S_j^L은 밀집 단말 영역이며, $\boldsymbol{S}^I$는 밀집 단말 영역들로만 이루어진 밀집 내부 영역이다.

마찬가지로 *d_highest*값이 B_{low}보다 작은 영역은 희소 내부 영역이며, 이에 대한 증명은 밀집 내부 영역에 대한 증명과 대칭적이므로 생략한다. □

```
Algorithm DB_Pruning
Input:   md_index : multidimensional index to be used for clustering
         N        : the number of objects stored in the database
         ρ        : clustering factor
Output:  Clusters
begin
1:   R_dense := { };
2:   NRO := (1-ρ)*N;
3:   curr_level := root level of md_index;
4:   while (NRO > 0 and curr_level < leaf level of md_index) begin
5:        Construct a list L of internal regions by reading all pages at the current level from md_index;
6:        Make a sorted list L_high (={R_1^high, R_2^high, ..., R_n^high}) by sorting L in the ascending order of d_highest;
7:        p := Find_Partition_Region(L_high, NRO);
8:        B_high := d_high(R_p^high);
9:        Make a sorted list L_low (={R_1^low, R_2^low, ..., R_n^low}) by sorting L in the ascending order of d_lowest;
10:       p := Find_Partition_Region(L_low, NRO);
11:       B_low := d_low(R_p^low);
12:       for each R_i^low whose d_low(R_i^low) > B_high begin        /* R_i^low is a dense internal region */
13:            R_dense := R_dense ∪ R_i^low;
14:            Prune the subtree whose root represents the internal page of R_i^low;
15:       end
16:       for each R_i^high whose d_high(R_i^high) < B_low begin      /* R_i^high is a sparse internal region */
17:            NRO := NRO - n(R_i^high);
18:            Prune the subtree whose root represents the internal page of R_i^high;
19:       end
20:       curr_level := curr_level + 1;
21:  end
22:  if(NRO > 0) begin
23:       Construct a list L of leaf regions by reading all remaining leaf pages from current md_index;
24:       Make a sorted list L_sort (={R_1, R_2, ..., R_m}) by sorting L in the ascending order of region density;
25:       p := Find_Partition_Region(L_sort, NRO);
26:            R_dense := R_dense ∪ {R_p, R_p+1, ..., R_m};
27:  end
28:  C := Find_Clusters(R_dense);
29:  Return C;
end
```

그림 6 밀도 기반 전지 알고리즘

예제 4: 그림 7은 밀도 기반 전지 알고리즘이 수행되는 과정을 예시한 것이다. 먼저 현재 계층의 색인 영역들이 그림 7(a)와 같다고 하자. 그리고, 각 영역은 그림 7(b)와 같은 정보를 갖는다고 하자. 그림 7(d)는 영역들의 *d_highest* 값으로 정렬한 리스트로서, ρ =0.9라 가정한 경우의 상위 한계값 B_{high}가 결정되는 것을 보여주고 있다. 또한, 그림 7(e)는 영역들의 *d_lowest* 값으로 정렬한 리스트로서, 하위 한계값 B_{low}가 결정되는 것을 보여주고 있다. 그림 7(d)와 (e)를 보면, 영역 R_1의 경우 *d_highest*가 B_{low}보다도 작은 값을 가지고 있음을 알 수 있다. 즉, 영역 R_1에 속한 모든 단말 영역들은 희소 단말 영역임을 나타내며, 따라서 영역 R_1을 루트로 하는 서브 트리는 전지가 가능하다. 반면, 영역 R_4와 R_8의 경우는 자신의 *d_lowest*가 B_{high}보다도 큰 값을 가지고 있음을 알 수 있다. 즉, 이들 영역에 속한 모든 단말 영역들은 밀집 단말 영역임을 나타내며, 따라서 이들 영역을 루트로 하는 서브 트리도 전지가 가능하다. 그림 7(c)는 전지된 결과를 영역으로 도시한 것이다. □

따름정리 1 밀도 기반 전지 알고리즘에 의해 구한 클러스터는 단말 대조 알고리즘으로 구한 클러스터와 동일하다.

증명: 밀도 기반 전지 알고리즘이 분기 한정 알고리즘이고 정리 3에서 한계값(bound)이 정확함을 증명하였으므로 이는 당연히 성립하며, 이에 대한 세부 증명 과정은 생략한다. □

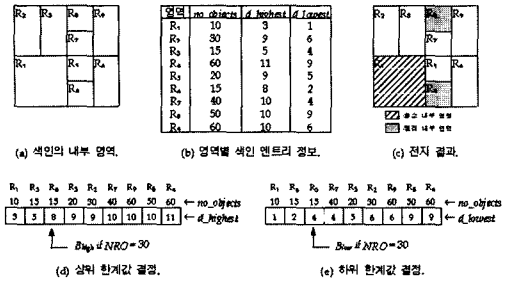


그림 7 밀도 기반 전지가 수행되는 예

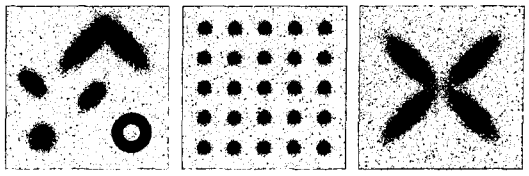
4. 성능 평가

본 절에서는 제안한 알고리즘과 가장 널리 알려진 클러스터링 알고리즘중의 하나인 BIRCH[6]와의 성능 평가 결과를 설명한다. 제4.1절에서는 성능 평가를 수행한 실험 데이타와 실험 환경을 설명한다. 다음으로 제4.2절에서는 각 알고리즘이 찾아낸 클러스터들의 정확성 실험 결과를 설명하고, 제4.3절에서는 제안하는 방법들과 BIRCH의 성능 평가 결과를 설명한다. 마지막으로, 제4.4절에서는 클러스터링 인수에 대한 감도 분석 결과를 설명한다.

4.1 실험 데이타 및 실험 환경

본 논문에서는 그림 8과 같은 세 가지 종류의 이차원 데이타 집합을 생성하여 실험에 사용하였다. 각 데이타 집합은 이차원 점 객체들로 구성되어 있으며, 각 차원은 도메인이 $[-2^{20}, 2^{20}-1]$인 정수값을 갖는다. 클러스터는 기본적으로 정규분포(normal distribution)를 사용하여 생성하였으며, 노이즈 객체는 균일분포(uniform distribution)를 사용하여 생성하였다. 클러스터의 크기와 모양은 정규분포의 파라메타인 표준편차와 두 차원의 상관계수를 달리하여 조정하였으며, 도메인 공간을 고려하여 표준편차는 2^{14}~2^{20}의 값을 사용하였다. 각 데이타 집합에 대한 자세한 생성 방법은 다음과 같다.

• DS1: 다섯개의 다양한 형태의 클러스터로 구성되어 있다. 원과 타원 형태의 세 개의 클러스터는 표준편차를 2^{15}로 하고 두 차원의 상관계수를 각각 0, 0.5, -0.5로 한 정규분포로 생성하였다. 타원 두 개가 맞닿은 형태의 클러스터는 표준편차가 2^{16}, 상관계수가 각각 0.8과 -0.8인 두 개의 정규분포로 생성하였다. 마지막으로, 도우넛 형태의 클러스터는 표준편차를 2^{16}로 하고 상관계수를 0으로 한 정규분포로 생성된 객체들 중에서 평균값에 가장 가까운 30%와 가장 먼 15%를 제거한 것이다.



(a) DS1 (b) DS2 (c) DS3

그림 8 실험 데이타 집합

• DS2: 동일한 개수의 객체들을 포함하는 25개의 클러스터로 이루어져 있다. 각 클러스터는 표준편차가 2^{14}이고, 두 속성간의 상관 계수가 0인 정규분포를 사용하여 생성하였으며, 25개의 클러스터는 이차원 공간상에서 그리드 형태로 규칙적으로 배열되어 있다.

• DS3: 방사형을 이루는 네개의 타원 형태의 클러스터로 이루어져 있다. 각 클러스터는 표준편차가 2^{16}, 상관계수가 0.8 또는 -0.8인 정규분포로 생성하였다.

DS1은 다양한 크기와 형태의 클러스터를 갖는 데이타 집합이고, DS2는 BIRCH에서 사용한 것과 유사한 형태의 데이타 집합으로서 정확성 및 성능 실험을 위한 것이며, DS3는 클러스터링 인수에 대한 감도 분석 실험을 위한 것이다. 각 데이타 집합에 속한 객체의 개수, 각 클러스터에 속한 객체의 개수, 노이즈 비율에 대해서는 다음 절에서 자세히 설명한다.

실험은 512M 바이트 메모리를 가진 SUN Ultra 60 워크스테이션에서 수행하였다. 데이타를 저장하는 다차원 색인 구조로는 MLGF[18]를 사용하고, 데이타 및 색인 페이지의 크기는 1024 바이트로 하였다.[1)]

4.2 정확성 실험 결과

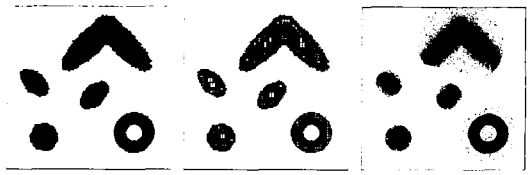
본 절에서는 제안한 알고리즘들과 BIRCH가 찾은 클러스터들에 대한 정확성 실험 결과를 설명한다. 실험을 위하여 각각 100만개의 객체를 포함하는 DS1과 DS2를 생성하였으며, 노이즈 비율은 전체 객체수의 8%로 하였다. 노이즈 비율이 8%이므로 제안한 알고리즘에서 클러스터링 인수 ρ는 0.92를 사용하였으며, 밀집 조각을 판별하는 ε의 값은 전체 객체수의 0.1%로 하였다. 그리고, BIRCH의 실행을 위한 입력 변수들은 저자들이 논문에서 제시한 기본값[6]을 사용하였다. 본 논문에서는 알고리즘으로 찾은 각 클러스터의 정확성을 정량적으로 표현하기 위하여 다음과 같이 클러스터 C의 정확도라는 성능 척도를 정의한다.[2)]

1) 페이지를 1024의 작은 크기로 한 이유는 높이가 큰 다차원 색인을 생성하여 전지 여부를 정확히 확인하기 위해서다. 페이지 크기로 4096을 사용하더라도 1024를 사용한 경우와 유사한 결과를 얻을 수 있다.

클러스터 C의 정확도=

$$\frac{\text{클러스터 } C\text{에 속한 객체들 중 주분포를 이루는 객체의 개수}}{\text{클러스터 } C\text{에 속한 전체 객체수}}$$

그림 9는 DS1에 대해 각 알고리즘들이 찾은 클러스터를 보여준다. 그림 9(a)는 LeafContrast(이하 *LC*라 한다)에 의해 찾은 클러스터 모양을 나타내고, 그림 9(b)는 DB_Pruning(이하 *DBP*라 한다)에 의해 찾은 클러스터를 나타내며, 그림 9(c)는 BIRCH에 의해 찾은 클러스터이다. 그림 9(a)를 보면, LC에 의해 찾은 클러스터들이 그림 8(a)의 원 데이타 집합의 클러스터들과 그 모양이 매우 유사함을 알 수 있는데, 이는 LC가 정확한 방법임을 의미한다. 그림을 보면, 클러스터 내부가 수 많은 작은 사각형으로 이루어져 있다. 이는 LC가 색인의 단말 영역들을 대상으로 클러스터링을 수행하기 때문이다. 즉, 클러스터 내부의 단말 영역들은 그 개수가 많은 반면 크기가 매우 작기 때문에, 이들이 표현된 사각형의 크기가 매우 작고 많은 것이다. 그림 9(b)를 보면, DBP에 의해 찾은 클러스터의 모양이 LC에 의해 찾은 클러스터의 모양과 동일함을 알 수 있는데, 이는 따름정리 1의 결과를 확인하는 것이다. 그림 9(b)와 그림 9(a)의 클러스터들을 비교해 보면, 그림 9(b)는 클러스터 내부의 많은 사각형들이 그림 9(a)의 사각형들보다 더 크다는 것을 알 수 있다. 이는 DBP에서 많은 밀도 기반 전지가 발생하였기 때문이다. 즉, 많은 내부 영역들이 밀집 영역으로 판별되었으며, 이들 영역은 단말 영역에 비해 그 크기가 커서 큰 사각형으로 나타난 것이다. 밀도 기반 전지와 관련된 성능 평가 결과는 제4.3절에서 자세히 설명한다. 마지막으로 그림 9(c) BIRCH의 결과를 보면, 찾은 클러스터의 형태가 모두 구형(spherical)임을 알 수 있다. 이는 BIRCH가 클러스터 요약 정보(*CF*)를 노드값으로 갖는 트리를 구성할 때,



(a) LC (b) DBP (c) BIRCH

그림 9 DS1에 대한 각 알고리즘의 클러스터링 결과

2) 클러스터 C에 속한 전체 객체수는 균일 분포를 통해 생성된 노이즈 객체들을 포함한 것이다.

표 2 DS1에 대한 각 알고리즘별 클러스터 정확도 비교

구 분	객체수				클러스터 정확도(%)	
	LC/DBP		BIRCH		LC/DBP	BIRCH
	전체*	원 집합**	전체	원 집합		
클러스터 1	398,638	390,935	404,076	388,644	98.1	96.2
클러스터 2	99,459	97,842	98,112	96,344	98.4	98.2
클러스터 3	99,517	97,862	97,608	95,913	98.3	98.3
클러스터 4	222,800	219,631	227,364	220,000	98.6	96.8
클러스터 5	99,402	97,556	97,457	95,904	98.1	98.4

* 전체 : 클러스터 C에 속한 총 객체수

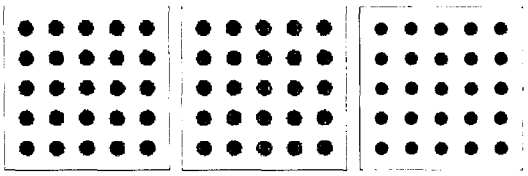
** 원 집합 : 클러스터 C에 속한 객체들 중 주 분포를 이루는 객체수

각 노드가 나타내는 클러스터는 중심으로부터의 반지름으로 모양이 결정되기 때문이다.

표 2는 DS1에 대해 찾은 클러스터들의 정확도를 정량적으로 나타낸 것이다. 앞서 언급한 바와 같이 제안한 알고리즘 LC와 DBP는 동일한 모양의 클러스터들을 찾아냈으며, 이에 따라 이들 알고리즘에 대한 정확도 또한 동일하게 나타났다. 표를 보면, 제안한 알고리즘들은 각 클러스터에 대하여 98.1%~98.6%의 고르고 높은 정확도를 나타내었다. 이는 제안한 알고리즘들이 원 데이타 집합의 클러스터들을 정확하게 식별함을 정량적으로 뒷받침하는 결과이다. BIRCH 또한 각 클러스터에 대한 정량적인 정확도는 96.2%~98.4%로 비교적 높게 나타났다. 이는 BIRCH가 찾은 구형의 클러스터내에 대부분의 밀집 영역들이 포함되었기 때문인데, 전체적으로는 LC 및 DBP가 약간 더 정확함을 알 수 있다.

그림 10의 (a)~(c)는 DS2에 대해 ρ를 0.92로 하고 ε을 0.001로 하였을 경우, 제안한 알고리즘과 BIRCH에 의해 찾은 클러스터를 보여준다. 그림 10(a)를 보면, LC에 의해 찾은 클러스터들이 그림 8(b)의 원 데이타 집합의 클러스터들과 모양에 있어 매우 유사함을 알 수 있다. 이는 클러스터의 개수가 많은 경우에도 LC가 정확한 클러스터링을 수행함을 의미한다. 그림 10(b)를 보면, DS1에서와 마찬가지로 DBP에서 밀도 기반 전지가 많이 이루어졌음을 알 수 있다. 그림 10(c)를 보면, BIRCH 역시 정확한 클러스터를 찾음을 알 수 있다. DS2에 대해 각 알고리즘들이 찾은 클러스터들의 정량적인 정확도는 LC와 DBP가 98.4%~98.7%, BIRCH가 98.6%~99.0%의 범위로 매우 높게 나타났다. 전체적으로는 BIRCH의 정확도가 다소 높게 나타났는데, 이는 그림 8(b)와 같이 모든 클러스터가 구형인 경우에는 클러스터 개수, 노이즈 기준 등과 같은 입력 변수를 최적값으로 지정해 주면 BIRCH가 매우 정확한 결과를 보임을 의미한다[6]. DS2의 각

클러스터별 정확도에 대한 세부 내용은 생략한다.



(a) LC (b) DBP (c) BIRCH

그림 10 DS2에 대한 각 알고리즘의 클러스터링 결과

4.3 성능 실험 결과

본 절에서는 제안한 알고리즘과 BIRCH에 대한 성능 평가 결과를 설명한다. 성능 평가의 척도로는 클러스터링 수행 과정에서 발생한 페이지 액세스 횟수를 사용한다. 실험을 위하여 각각 10만개, 100만개, 1,000만개의 객체를 포함하는 DS1 및 DS2를 생성하였으며, 노이즈 비율은 전체 객체수의 8%로 하였다. 노이즈 비율이 8%이므로 알고리즘에서의 클러스터링 인수 ρ는 0.92를 사용하였으며, 밀집 조각을 판별하는 ε의 값은 전체 객체수의 0.1%로 하였다. 그리고, BIRCH의 실행을 위한 입력 변수들은 저자들이 논문에서 제시한 기본값[6]을 사용하였다.

그림 11은 서로 다른 크기의 DS1에 대한 각 알고리즘의 페이지 액세스 횟수를 나타낸다. 그림 11을 보면, BIRCH의 페이지 액세스 횟수가 가장 많음을 알 수 있다. 이는 BIRCH가 데이타베이스 전체(본 논문의 경우 다차원 파일 구조의 데이타 페이지 전체)를 한 번 액세스하여야 하기 때문이다. 반면에, 데이타 페이지를 액세스하지 않고 다차원 파일 구조의 색인 페이지만을 액세스하는 LC는 BIRCH에 비해 페이지 액세스 횟수를 크게, 약 1/15 수준으로 줄인 것으로 나타났다. 객체수가 증가하더라도 LC와 BIRCH의 페이지 액세스 횟수 비율은 일정하게 유지됨을 알 수 있는데, 이는 색인 크기(페이지 개수)가 전체 데이타베이스의 크기에 비례하기 때문이다. 다음으로, DBP에서는 밀도 기반 전지가 일어나기 때문에, 페이지 액세스 횟수를 LC보다도 더욱 줄였음을 알 수 있다. 특히, 객체수가 증가함에 따라 전지되는 페이지 개수의 비율이 높아짐을 알 수 있다. 이는 객체의 수가 많아져 밀집도가 높아지면, 전지되는 색인 검색 공간이 많아지게 되고, 결과적으로 클러스터링 성능 향상 효과가 더욱 커짐을 의미한다. DS1에 대한 실험 결과를 종합하면, 제안한 DBP는 BIRCH에 비해서 1/26에서 최대 1/187까지 페이지 액세스 횟수를 줄인 것으로 나타났다. 따라서, 제안한 DBP는 대용량 데이타베이스에 보다 적합한 클러스터링 방법이라 할 수 있다.

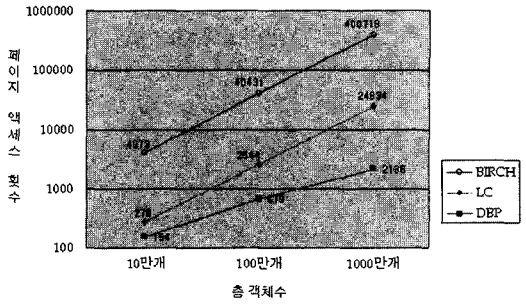


그림 11 서로 다른 크기의 DS1에 대한 각 알고리즘의 페이지 액세스 횟수

표 3은 서로 다른 크기의 DS1에 대한 DBP의 전지 결과를 수치로 보여준다. 표 3에 따르면, DBP는 전체 색인 페이지 중에서 44.2%~91.4%를 전지하며, 전체 객체 중에서 55.7%~93.6%를 전지하는 것으로 나타났다.

그림 12는 서로 다른 크기의 DS2에 대한 각 알고리즘의 페이지 액세스 횟수를 나타낸다. DS2에 대해서도 DS1에서와 유사한 성능 향상을 보임을 알 수 있다. DBP는 BIRCH에 비해서 1/13에서 최대 1/149까지 페이지 액세스 횟수를 줄인 것으로 나타났다. 그리고, 색인 페이지와 객체수에 대한 전지 비율은 각각 25.6%~89.3%와 49.3%~92.3%로 나타났다.

표 3 서로 다른 크기의 DS1에 대한 DBP의 전지 결과

전체 객체수	전체 색인 페이지수	전지된 색인 페이지수(%)	전지된 객체수(%)
100,000	276	122(44.2)	55,695(55.7)
1,000,000	2,544	1,865(73.3)	803,305(80.3)
10,000,000	24,834	22,698(91.4)	9,361,793(93.6)

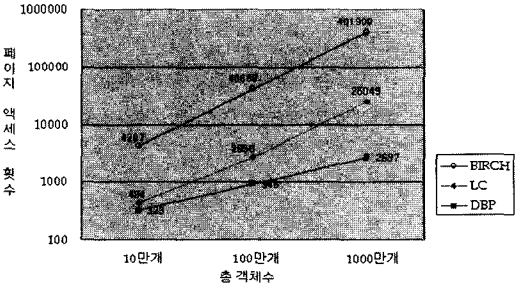


그림 12 서로 다른 크기의 DS2에 대한 각 알고리즘의 페이지 액세스 횟수

4.4 클러스터링 인수에 대한 감도 분석

마지막으로 본 절에서는 클러스터링 인수 즉, 데이타 집합의 노이즈 수준에 따른 클러스터링 결과를 분석한다. 제안한 알고리즘에서 노이즈 수준은 클러스터링 인수(ρ)에 의해 결정된다. 실험을 위하여 10만개의 객체를 포함하는 DS3를 생성하였으며, 밀집 조각을 판별하는 ε의 값은 전체 객체수의 0.1%로 하였다.

그림 13의 (a)~(c)는 DS3에 대해 ρ를 변화시켜가며 수행한 클러스터링 결과를 나타낸 것이다. 그림 13(a)는, ρ=0.70 즉, 노이즈 비율을 전체 데이타 개수의 30%로 한 경우의 클러스터링 결과이다. 객체들이 밀집되어 있지 않는 영역들을 제거한 결과 총 4개의 클러스터를 식별하였음을 알 수 있다. 그림 13(b)는, ρ=0.80인 경우의 클러스터링 결과이다. 그림 13(a)에서 제거되었던 영역들 중에서 상대적으로 밀도가 높은 영역들이 밀집 영역으로 추가되어 상하 클러스터간의 영역들이 이행적 인접함으로써, 결과적으로 총 2개의 클러스터를 식별하였음을 알 수 있다. 그림 13(c)는, ρ=0.85인 경우의 클러스터링 결과이다. 마찬가지로, 그림 13(b)에서 제거되었던 영역들 중에서 상대적으로 밀도가 높은 영역들이 밀집 영역으로 추가되어 좌우 클러스터간의 영역들이 이행적 인접함으로써, 결과적으로 총 1개의 클러스터를 식별하였음을 알 수 있다.

(a) ρ=0.70 (b) ρ=0.80 (c) ρ=0.85

그림 13 노이즈 수준에 따른 감도 분석 결과

5. 결 론

본 논문에서는 대용량 데이타베이스에 대해 효율적으로 적용 가능한 새로운 계층 클러스터링 알고리즘을 제안하였다. 제안한 방법은 계층 트리를 루트로부터 아래로 검색해 가며 클러스터링을 수행하는 하향식 기법을 사용한다. 그리고, 실험을 통하여 제안한 방법이 기존의 상향식 계층 클러스터링 방법인 BIRCH[6]에 비해 유사한 정확도를 가지면서도 훨씬 우수한 성능을 나타냄을 보였다.

먼저, 하향식 클러스터링 기법을 적용하기 위하여 데이타베이스의 다차원 색인이 갖는 클러스터링 성질을 이용하여 전체 색인 공간을 분할하는 영역 대조 분할 개념을 제시하고, 이를 기반으로 클러스터를 정형적으로 정의하였다. 정의한 클러스터는 색인의 모든 단말 페이지에 포함된 각 엔트리들이 나타내는 영역들을 밀도가 높은 밀집 영역과 밀도가 낮은 희소 영역으로 분할하고, 밀집 영역들을 대상으로 인접한 영역들의 집합을 구한 것이다. 이렇게 함으로써 데이타베이스의 모든 객체들을 액세스하지 않고 색인 정보만을 이용하여 클러스터의 모양과 크기를 결정할 수 있다. 그리고, 이와 같이 정의한 클러스터를 찾는 직관적인 알고리즘으로서 색인의 단말 영역들을 대상으로 클러스터링을 수행하는 단말 대조 알고리즘(간단히 *LC*라 한다)을 제안하였다. 또한, LC에 의해 찾은 클러스터가 정의한 클러스터와 동일함을 정리 2를 통하여 증명하였다.

다음으로, 밀도 기반 전지 개념을 이용하여 불필요한 검색 공간을 전지함으로써, 클러스터링 성능을 향상시키는 밀도 기반 전지 알고리즘(간단히 *DBP*라 한다)을 제안하였다. DBP는 분기 한정 알고리즘으로서 이를 위한 효율적인 한계값을 제시하고, 이 값이 정확함을 정리 3을 통하여 증명하였다.

제안하는 알고리즘의 성능을 평가하기 위해 데이타의 종류와 크기를 달리하여 많은 실험을 수행하였다. 실험 결과, 제안하는 방법은 정확성 측면에서 BIRCH와 유사하거나 우수한 것으로 나타났다. 그리고, 성능 측면에서는 전체적으로 BIRCH보다 훨씬 우수하였으며, 특히 밀도 기반 전지에 의한 성능 향상 효과가 큰 것으로 나타났다. 대용량 데이타 집합(객체수 1,000만개)에 대해 실험을 수행한 결과, LC는 BIRCH에 비해 페이지 액세스 횟수를 1/13~1/16로 줄였으며, 특히 DBP는 BIRCH에 비해 페이지 액세스 횟수를 최대 1/187로 크게 줄인 것으로 나타났다.

본 논문에서 제안한 다차원 색인을 이용한 하향식 클러스터링 기법은 대용량 데이타베이스에 대한 클러스터링 성능을 크게 향상시키는 새로운 기법으로서, 일반 데이타베이스 응용에 실용적으로 적용 가능할 것으로 믿는다.

참 고 문 헌

[1] M. S. Chen, J. Han, and P. S. Yu, "Data Mining: An Overview from a Database Perspective," *IEEE Trans. on Knowledge and Data Engineering*, Vol. 8, No. 6, pp. 866-883, Dec. 1996.

[2] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, *"A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,"* In *Proc. the 2nd Int'l Conf. on Knowledge Discovery and Data Mining(KDD)*, Portland, Oregon, pp. 226-231, Aug. 1996.

[3] S. Guha, R. Rastogi, and K. S. Shim, *"CURE: An Efficient Clustering Algorithm for Large Databases,"* In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Seattle, Washington, pp. 73-84, June 1998.

[4] R. T. Ng and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining," In *Proc. the 20th Int'l Conf. on Very Large Data Bases*, Santiago, Chile, pp. 144-155, Sept. 1994.

[5] W. Wang, J. Yang, and R. Muntz, "STING: A Statistical Information Grid Approach to Spatial Data Mining," In *Proc. the 23rd Int'l Conf. on Very Large Data Bases*, Athens, Greece, pp. 186-195, Aug. 1997.

[6] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Montreal, Quebec, Canada, pp. 103-114, June 1996.

[7] M. Breunig, H. P. Kriegel, P. Kroger, and J. Sander, "Data Bubbles: Quality Preserving Performance Boosting for Hierarchical Clustering," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Santa Barbara, California, pp. 79-90, May 2001.

[8] V. Ganti, R. Ramakrishnan, J. Gehrke, A. Powell, and J. French, "Clustering Large Datasets in Arbitrary Metric Spaces," In *Proc. the 15th Int'l Conf. on Data Engineering(ICDE)*, Sydney, Australia, pp. 502-511, Feb. 1999.

[9] M. Ankerst, M. Breunig, H. P. Kriegel, and J. Sander, "OPTICS: Ordering Points To Identify the Clustering Structure," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Philadelphia, Pennsylvania, pp. 49-60, June 1999.

[10] G. Karypis, E. H. Han, and V. Kumar, "Chameleon: Hierarchical Clustering Using Dynamic Modeling," *IEEE Computer*, Vol. 32, No. 8, pp. 68-75, Aug. 1999.

[11] M. Ester, H. P. Kriegel, and X. Xu, "Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification," In *Proc. the 4th Int'l Symp. on Large Spatial Databases(SSD)*, Portland, Maine, pp. 67-82, Aug. 1995.

[12] Erich Schikuta, "Grid-clustering: An efficient hierarchical clustering method for very large data sets," In *Proc. the 13th Int. Conf. on Pattern Recognition*, Vienna, Austria, Vol. 2, pp. 101-105, Oct. 1996.

[13] G. Sheikholeslami, S. Chatterjee, and A. Zhang, "WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases," In *Proc. the 24th Int'l Conf. on Very Large Data Bases*, New York City, New York, pp. 428-439, Aug. 1998.

[14] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & Sons, 1990.

[15] J. H. Lee, Y. K. Lee, K. Y. Whang, and I. Y. Song, *"A Region Splitting Strategy for Physical Database Design of Multidimensional File Organizations,"* In *Proc. the 23rd Int'l Conf. on Very Large Data Bases*, Athens, Greece, pp. 416-425, Aug. 1997.

[16] C. R. Palmer and C. Faloutsos, "Density Biased Sampling: An Improved Method for Data Mining and Clustering," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Dallas, Texas, pp. 82-92, May 2000.

[17] J. Nievergelt, H. Hinterberger, and K. C. Sevcik, "The Grid File: An Adaptable, Symmetric Multikey File Structure," *ACM Trans. on Database Systems*, Vol. 9, No. 1, pp. 38-71, Mar. 1984.

[18] K. Y. Whang and R. Krishnamurthy, Multilevel Grid Files, IBM Research Report RC11516, 1985.

[19] Kyu-Young Whang, Sang-Wook Kim, and Gio Wiederhold, "Dynamic Maintenance of Data Distribution for Selectivity Estimation," *The VLDB Journal*, Vol. 3, No. 1, pp. 29-51, 1994.

부 록

정리 1의 증명: 전체 색인 영역 $\boldsymbol{R}$이 서로 소인 영역들의 집합 $\{R_1, ..., R_m\}$으로 이루어져 있고, $\boldsymbol{R}$에서의 제거 가능한 객체수를 NRO라 하자. 이 집합 $\boldsymbol{R}$을 영역 대조 분할하는 방법을 제시함으로써 이 정리를 증명한다. 우선 모든 영역들을 밀도를 키로 한 오름차순으로 정렬하고, 이 때의 순서를 이라 하자. 이제 다음과 같은 식 (1)을 만족하는 p를 찾는다.

$$(\sum_{i=1}^{p-1} n(R_{x_i}) \le NRO) \wedge (\sum_{i=1}^{p} n(R_{x_i}) > NRO) \qquad (1)$$

$\boldsymbol{R}$의 총 객체수 $N = \sum_{i=1}^{m} n(R_{x_i})$이므로, 식 (1)을 만족하는 $p(1 \le p \le m)$는 항상 존재함을 알 수 있다. 그러므로, $\boldsymbol{R_S} = \{R_{x_1}, ..., Rx_{p-1}\}$, $\boldsymbol{R_D} = \{R_{x_p}, ..., R_{x_m}\}$으

로 삼으면, 가정에 의해 정의 4의 조건 1과 조건 2는 당연히 만족한다. 또, 식 (1)에서 R_{x_p}는 정의 4의 조건 3의 R_p가 되며, R_D에서의 최저 밀도 영역이 된다. 따라서, 영역 기준 분할전략을 사용하는 다차원 색인 영역들의 집합에 대해서는 항상 정의 4의 세 가지 조건을 만족하는 분할이 가능하다. □

황 재 준

1980년 2월 서울대학교 전기공학과 학사. 1982년 2월 서울대학교 제어계측공학과 석사. 1996년 3월 ~ 현재 한국과학기술원 전자전산학과 전산학전공 박사과정. 1988년 4월 ~ 현재 국방과학연구소 책임연구원. 관심분야는 데이타 마이닝, 공간 인덱싱

문 양 세

1991년 2월 한국과학기술원 과학기술대 전산학과 학사. 1993년 2월 한국과학기술원 전산학과 석사. 2001년 8월 한국과학기술원 전자전산학과 전산학전공 박사. 1991년 3월 ~ 2001년 6월 현대전자산업㈜ 통신연구소 선임연구원. 2001년 7월 ~ 2002년 2월 ㈜현대시스콤 통신연구소 선임연구원. 2002년 2월 ~ 현재 ㈜인프라밸리 기술연구소 수석연구원. 관심분야는 Data Mining, Knowledge Discovery, Access Methods, Mobile & Wireless Communication Systems, Network Communication Systems

황 규 영

1973년 서울대학교 전자공학과 졸업(B.S.). 1975년 한국과학기술원 전기 및 전자학과 졸업(M.S.). 1982년 Stanford University(M.S.). 1983년 Stanford University(Ph.D.). 1975년 ~ 1978년 국방과학연구소(ADD), 선임연구원. 1983년 ~ 1990년 IBM T.J. Watson Research Center, Research Staff Member. 1992년 ~ 1994년 한국정보과학회 데이타베이스 연구회(SIGDB) 운영위원장. 1995년 한국정보과학회 이사 겸 논문지 편집위원장. 1999년 ~ 2000년 한국정보과학회 부회장. 1990년 ~ 현재 Editor: The VLDB Journal. 1991년 ~ 1995년 Editor: Distributed and Parallel Databases: An International Journal. 1994년 ~ 현재 Editor: International Journal of Geographical Information Systems. 1990년 ~ 1993년 Associate Editor: IEEE Data Engineering Bulletin. 2002년 ~ 현재 Editor: IEEE Transactions on Knowledge and Data Engineering. 1998년 ~ 2004년 Trustee : VLDB Endowment. 1999년 ~ 2005년 Steering Committee Member, DASFAA. 1990년 ~ 현재 한국과학기술원 전자전산학과 전산학과 교수. 1999년 ~ 현재 첨단정보기술연구센터(한국과학재단지정 우수연구센터) 소장. 관심분야는 데이타베이스 시스템, 멀티미디어, GIS