점진적 프로젝션을 이용한 고차원 클러스터링 기법
(High-Dimensional Clustering Technique using Incremental Projection)

이 혜 명 † 박 영 배 ††
(Hye-Myung Lee) (Young-Bae Park)

요 약 대부분의 클러스터링 알고리즘들은 고차원 공간에서 성능이 급격히 저하되는 경향이 있다. 더욱이, 고차원 데이타는 상당한 양의 잡음 데이타를 포함하고 있으므로 알고리즘의 추가적인 효과성 문제를 야기한다. 그러므로 고차원 데이타의 구조와 특성을 지원하는 적합한 클러스터링 기법이 개발되어야 한다.

본 논문에서는 선형변환 프로젝션을 이용한 클러스터링 알고리즘 CLIP을 제안한다. CLIP은 고차원 클러스터링의 효율성 및 효과성 문제를 극복하기 위해 개발되었으며, 클러스터 형성에 밀접하게 연관된 부분공간에서 클러스터를 탐사하는 기법이다. 알고리즘의 주요 사상은 각 1차원적 부분공간에서의 클러스터링에 기본을 두고 있지만, 점진적인 프로젝션을 이용하여 고차원 클러스터를 탐사할 뿐만 아니라 연산을 획기적으로 줄인다. CLIP의 성능을 평가하기 위해 합성 데이타를 이용한 일련의 실험을 통하여 효율성 및 효과성을 증명한다.

Abstract Most of clustering algorithms tend to degenerate rapidly on high dimensional spaces. Moreover, high dimensional data often contain a significant amount of noise, which causes additional ineffectiveness of algorithms. Therefore, it is necessary to develop algorithms adapted to the structure and characteristics of the high dimensional data.

In this paper, we propose a clustering algorithm, CLIP, using the projection. The CLIP is designed to overcome efficiency and/or effectiveness problems on high dimensional clustering and it is the method to find clusters in the corresponding subspace correlated closely. The key idea in the algorithm is based on clustering on each one dimensional subspace but we use the incremental projection to recover high dimensional cluster and to reduce the computational cost significantly at same time. To evaluate the performance of CLIP, we demonstrate its efficiency and effectiveness through a series of experiments on synthetic data sets.

1. 서 론

데이타 마이닝을 위한 방법론 중에서 클러스터링은 데이타 집합에 대하여 유사한 특징을 가진 객체들을 집단화하는데 사용되는 매우 유용한 분석방법이다. 클러스터링에 의한 분석은 데이타 마이닝을 위한 우선적인 방법이 될 수 있으며, 독자적인 도구로서 심도있는 분석을 하거나 다른 알고리즘 수행을 위한 전처리 과정으로 사용할 수 있다[1]. 또한 클러스터링은 데이타베이스 분야에서도 유사성 검색, 고객 분류, 경향 분석 등에서 폭넓게 응용된다.

이에 따라 많은 클러스터링 알고리즘들이 개발되고 있으나, 잘 알려진 대부분의 알고리즘들은 고차원 데이타 공간에서 클러스터를 탐사하는데 실패하는 경향이 있다. 이와 같은 현상은 주로 고차원 공간의 데이타 점들이 갖는 자체의 희소성(sparsity)에 기인하며, 차원 전체가 클러스터 형성에 관련되지 않을 수 있다는 것이다[2,3].

고차원 공간에서 하나의 데이타 점에 해당하는 데이타 레코드는 일반적으로 수십 개의 애트리뷰트를 갖으며, 각 애트리뷰트의 도메인은 매우 방대하다. 이러한 고차원 공간에서 데이타 점들의 평균 밀도로써 클러스

† 정 회 원 : 경문대학 컴퓨터정보과 교수
hmlee@kmc.ac.kr
†† 종신회원 : 명지대학교 컴퓨터공학과 교수
parkyb@mju.ac.kr
논문접수 : 2001년 5월 9일
심사완료 : 2001년 9월 24일

터를 찾는 것은 큰 의미가 없다. 이 문제를 보다 복합적으로 보면, 애트리뷰트의 차원들 중에서 많은 차원과 이들 차원의 조합은 잡음(noise)이나 균등하게 분포된 값을 가질 수 있으므로 데이타의 모든 차원을 고려하는 거리 함수는 비효율적이다. 이를 해결하기 위한 한가지 방법으로 클러스터 형성에 연관성 있는 차원을 선택하고 해당하는 부분차원에서 클러스터를 찾고자 하는 연구가 진행되고 있다.

관련성 있는 차원이나 공간을 고려하는 기존의 알고리즘으로는 CLIQUE[4], PROCLUS[2] 등이 있으며, 이들은 고차원 공간에서 데이타 점들은 전체차원의 부분집합 즉 일부차원에 대해 보다 효과적으로 클러스터링할 수 있다는 개념에 근거한다. CLIQUE는 부분차원 클러스터링의 첫 번째 연구로서 큰 의의가 있으나, 그리드 방식의 근본적인 문제점인 그리드 셀의 수가 지수적으로 증가함에 따른 부작용으로서 공간 및 시간적인 효율성의 저하를 초래한다. 또한 데이타 점들을 서로 소인 집합으로 분할하기 어렵기 때문에 엄밀한 정의의 클러스터 탐색에는 한계가 있다. PROCLUS는 부분차원에 존재하는 클러스터를 효과적으로 찾을 수 있는 접근방법이지만, 문제점으로는 주로 특정 차원에 국한된 희소 데이타 분석에만 용이할 수 있으며, 또한 고차원 공간의 대량의 데이타에 대해 최상의 medoid를 선택하는데 어려움이 따른다.

본 논문에서는 대용량의 고차원 데이타 공간에서 클러스터링의 효율성 및 효과성을 보장하기 위한 접근방법으로 CLIP(CLustering based on Incremental Projection) 알고리즘을 제안한다. CLIP은 점진적인 프로젝션을 통해 고차원 데이타를 선형적으로 처리함으로써 고차원 문제를 해결하고자 하는 클러스터링 기법이다. 또한 클러스터 형성에 밀접하게 연관성이 있는 차원 및 영역을 식별할 수 있는 부분차원 클러스터링 기법이다. 부분차원 기반의 클러스터링은 고차원 클러스터를 분석하기 위한 목적으로 부분차원의 명세가 요구될 때 이를 지원할 수 있으며, 입력 데이타 레코드에 누락된 값이 있어도 이를 허용할 수 있는 장점이 있다.

2. 관련 연구

고차원 데이타베이스에서 자동화된 클러스터링은 매우 중요한 문제이며 고차원 데이타를 적용할 수 있는 다양한 클러스터링 기법이 연구되고 있다. 클러스터링을 위한 대표적 접근방법으로는 분할기반 기법, 계층기반 기법, 밀도기반 기법으로 분류할 수 있다.

CLARANS[5]와 같은 분할 알고리즘은 데이타베이스를 k개의 클러스터 즉 클러스터의 무게중심(k-means) 또는 클러스터의 대표 객체(k-medoid)에 의하여 분할(partition)을 형성한 후 각 객체는 가장 가까운 클러스터에 할당된다. 계층적 클러스터링은 통상 트리 구조에 의하여 데이타베이스를 몇 단계의 분할로 분해한다. 이러한 계층적 알고리즘은 지식탐사에서 매우 효과적이지만, 트리를 생성하는 비용 때문에 대용량 데이타베이스에 대해서는 비현실적이다. 보다 효율적인 기법으로는 지역성(locality)을 고려한 밀도기반의 클러스터링으로서 인접한 데이타 요소들을 지역적 조건을 기반으로 하여 클러스터로 그룹화하며 데이타베이스를 한번만 스캔하는 클러스터링이 가능하다. DBSCAN[6]은 대표적인 밀도기반 클러스터링 기법이다.

보다 최적화된 클러스터링을 위한 기법들이 제안되었다. 예로서 효율성을 증대하기 위해 일정한 그리드(grid)를 사용하는 DENCLUE[7], 클러스터-특징 트리에 기반한 BIRCH[8], 추가적 통계정보를 포함하며 quadtree와 닮은 구조를 사용하는 STING[9] 등이 있다.

그러나 이상에서 열거한 대부분의 클러스터링 알고리즘들은 소위 "차원의 저주[10]" 현상에 기인하여 고차원에서 효과적 및 효율적으로 수행하지 못한다. 기존의 알고리즘들이 고차원 공간에서 실패하는 원인은 데이타 점들이 갖는 고유의 희소성 때문이다. 즉 고차원 공간에서 차원의 전체가 주어진 클러스터에 관련되지 않을 수 있다는 개념으로서 이것을 다루는 방법 중 하나가 연관된 차원을 고르는 것으로 해당하는 부분차원에서 클러스터를 탐사한다. 다음의 CLIQUE와 PROCLUS는 대표적인 부분차원 클러스터링 기법이다.

CLIQUE(CLustering In QUEst)는 다음의 세 단계로 수행된다. 첫 번째는 클러스터를 포함하는 부분차원을 식별하는 단계로서 연관 규칙을 위한 Apriori 알고리즘[11]과 유사한 bottom-up 알고리즘을 이용하여 부분차원에 존재하는 밀집단위(dense unit)를 찾는다. 두 번째는 클러스터를 탐사하는 단계로서, 첫 번째 단계의 결과인 밀집단위들의 집합을 입력으로 하여 분할된 클러스터 집합을 출력한다. 세 번째 단계에서는 클러스터의 최소 명세(minimal description)를 생성한다. CLIQUE는 데이타의 흥미있는 특성을 발견할 수 있는 접근방법이지만, 데이타 점들을 서로 소인 집합으로 분할하기 어렵기 때문에 엄밀한 정의의 클러스터 탐색에는 한계가 있다. 또한 주어진 밀집영역에 대하여, 저차원 부분공간에서 그것의 모든 부분차원들은 또한 밀집하다고 조사되므로 조사된 밀집영역 사이에는 큰 오버랩이 존재할 수 있다.

PROCLUS(PROjected CLUStering)는 고차원 공간에서 클러스터를 탐색하는데 프로젝트된 클러스터링 개

넘을 논의한 알고리즘으로서, 데이타 점 및 차원을 기반으로 클러스터를 산출한다. PROCLUS에서 제안한 프로젝트된 클러스터링은 우선, CLARANS에서 제안한 mediod 탐색기법을 이용하여 클러스터와 차원의 적당한 집합을 찾는다. 그 다음, 찾아진 각 mediod와 연관된 차원의 집합을 찾기 위하여 지역성 분석(locality analysis)기법을 사용한다. PROCLUS는 부분차원에 존재하는 클러스터를 효과적으로 찾을 수 있는 접근방법이지만 주로 특정 차원에 국한된 희소 데이타 분석에만 용이할 수 있으며, 또한 고차원 공간에서 대량의 데이타에 대해 최상 medoid를 선택하는데 어려움이 따른다.

3. 제안하는 클러스터링 기법

CLIP은 부분차원 클러스터링 기법으로서 클러스터 형성에 연관된 차원을 선택한 후 해당하는 후보공간에서 클러스터를 탐사한다. 따라서 본래의 주어진 전체 데이타 공간에서보다 더 나은 클러스터링을 위해 고차원 데이타 공간의 부분공간(subspaces) 또는 부분차원(subdimensions)을 자동적으로 식별하는데 관심을 둔다. 특히 CLIP은 부분차원의 의미를 원래 차원의 선형조합 등 새로운 차원을 이용하지 않고 원래 데이타 공간의 부분공간으로 그 범위를 제한하는데, 이러한 제한은 클러스터링 결과를 보다 간단하고 이해하기 쉽게 표현할 수 있으므로 매우 중요하다.

CLIP의 주요 사상은, 고차원 데이타를 저차원으로 변환하기 위해 프로젝션을 이용하는데 있다. 즉 k차원 데이타 공간에서 각 차원의 밀도에 근거하여 한 차원씩 점진적으로 프로젝션을 하면서 클러스터를 탐사하는 것이다. 또한 CLIP은 알고리즘의 효율성 및 효과성을 향상시키기 위해 클러스터링의 고전적인 접근방법의 기본 개념을 혼용하고 있으며 그 내용은 다음과 같다. 첫째, CLIP은 클러스터링을 위해 밀도-기반 접근방법을 사용하여 영역을 결정한다. 즉 클러스터는 주변 영역보다 데이타 점의 밀도가 높은 영역을 의미한다. 둘째, CLIP은 데이타 점들의 밀도를 간단히 산정하기 위해 데이타 공간을 일정한 간격으로 나누고 분할된 각 단위 안에 놓여진 점들의 수를 찾는 그리드-기반 접근방식을 사용한다. 다음의 그림 1은 CLIP 알고리즘의 전체적인 흐름을 도시한 것이다.

3.1 선형변환 프로젝션

CLIP은 전체 k차원 공간의 데이타 집합에 대해 프로젝션을 이용하여 선형변환(linear transformation)한다. 한 차원을 프로젝션하여 기준밀도를 초과하는 밀집영역을 찾은 뒤, 그 밀집영역에 종속적인 그 다음 차원의 밀집영역을 찾아내는 과정을 최종 k차원까지 반복적으로 수행한다. 프로젝션하는 영역의 선택은 밀도에 근거하므로 단계가 증가함에 따라 잡음 데이타 수는 점차 감소하게 된다. 이때 프로젝션하는 차원의 순서는 임의적일 수 있는데 차원의 중요도에 따라 우선 순위를 부여하여 순서를 결정한다.

(정의 1) 축-평행한 프로젝션

전체 k차원 공간 R^k의 데이타 집합에 대한 l-축으로의 프로젝션 P_l을 다음과 같이 정의한다.

$P_l : R^k \rightarrow R^k$; $P_l(x^{(1)}, x^{(2)}, \cdots, x^{(l)}, \cdots, x^{(k)}) = \hat{x}^{(l)}$, $1 \le l \le k$

여기서 $\hat{x}^{(l)} = (0, \cdots, 0, x^{(l)}, 0, \cdots, 0)$ 이다. 즉, l-축을 제외한 모든 항은 0(zero)으로 한다.

CLIP에서 사용하는 프로젝션의 의미는 다음의 (정의 1)과 같으며, 특히 그림 2와 같이 축-평행한 프로젝션(axis parallel projection)을 하는데 특정 축에 해당하는 차원을 제외한 다른 모든 차원의 값은 0으로 간주하고 데이타를 조사하는 것을 의미한다.

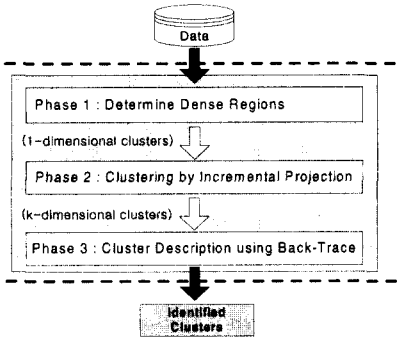

그림 1 CLIP의 개요

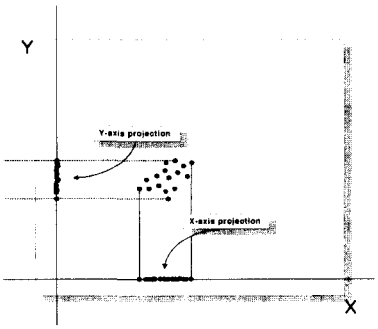

그림 2 축-평행한 프로젝션

고차원 데이타에 대해 차원의 전체를 고려하는 대신, 특정 좌표축에 대한 선형 변환인 프로젝션을 이용하는 데 있어서 얻는 장점은 다음과 같다[12]. 첫째, 데이타를 프로젝션하면 데이타 분포에 관한 더 많은 정보를 포함하고 있으며 본래의 데이타 공간에서 보다 작은 공간에 표현할 수 있다. 둘째, 저차원 공간으로 프로젝션하면 전체 공간에서 보다 데이타들의 희소성이 감소하여 더욱 조밀하게 분포하게 되며, 따라서 기존의 클러스터 탐사 기법들 보다 효과적으로 수행할 수 있다.

모집단 $X \subset R^k$의 l축의 프로젝션은 $X_l := P_l(X) = \{P_l(\boldsymbol{x}): \boldsymbol{x} \in X\}$이다. 만일 X의 평균이 $\boldsymbol{\mu} = (\mu_1, \mu_2, \cdots, \mu_k)$이면, X_l의 평균은 $\mu_l = P_l(\boldsymbol{\mu})$이다. 이와 같이 프로젝션은 본래 고차원 공간의 데이타 집합에 관한 통계적 물리량을 보존한다.

3.2 각 차원에서 클러스터의 결정

클러스터링 있어서 중요하게 고려할 사항은 각 차원에서의 밀집영역 즉 1차원적 클러스터를 찾는데 있으며 CLIP이 사용하며 방법은 다음과 같다.

데이타 공간의 각 차원을 겹침이 없는 단위(u)로 분할하는데, 그 단위들은 모든 차원을 일정한 ξ간격으로 나누어 얻어진다. CLIP은 각 차원에 대해 다른 ξ값을 갖도록 허용하는데, 이것은 서로 다른 부분공간에서 단위의 밀도를 조사할 때 공간의 상대적인 크기에 따라 밀도 임계값 τ를 조절할 수 있기 때문이다. 각 단위 u는 각 애트리뷰트에서 한 간격의 교차점으로서 각 차원은 집합 $\{u_1, u_2, \ldots, u_d\}$로 정의할 수 있으며, 하나의 간격은 $ui = [li, hi)$로 표현한다. 하나의 데이타 점 $v = \langle v_1, v_2, \ldots, v_d \rangle$는 하나의 단위 $u = \{u_1, u_2, \ldots, u_d\}$에 포함된다. 이때 한 단위의 선택(selectivity) 여부는 단위 안에 포함된 모든 점들의 수로 정의한다. 즉 "단위 u가 밀집(*dense*)하다"는 것은 ($selectivity(u) \geq \tau$)을 만족할 때임을 뜻한다. 클러스터를 형성하는 임의 부분차원의 밀집영역은 연결된 밀집단위(dense unit)들의 최대 집합(maximal set)으로 정의한다. 만일 2개의 밀집단위들 u_1, u_2가 하나의 공통된 면을 갖는다면, 2개의 단위들 u_1, u_2는 연결되어 있다(*connected*)고 한다.

그러나 결정된 밀집단위의 개수는 예상외로 많을 수 있으므로 후보가 되는 밀집 단위의 모임을 선택해야 한다. 의미있는 밀집영역을 결정하기 위해 밀집영역에 속한 점들의 수를 이용하며 다음과 같이 계산한다.

$xR_j = \sum_{u_i \in R_j} count(u_i)$ ($count(u_i)$는 ui안에 속한 데이타 점들의 수). 이때 xR_j는 밀집영역 R_j의 중요도로 참조할 수 있다. 즉 상대적으로 데이타 점들을 많이 포함하고 있는 밀집영역은 선택하고 나머지는 제외시킨다. 다음의 그림 3은 이상의 과정을 의미하는 밀집영역 결정 알고리즘이다.

```
/* Input: Data_rec, τ
   Output: Found (1, 0, null) */
 procedure Find_Dense(Data_rec) {
  Scan Data_rec; // 입력 데이타 집합 조사
  Evaluate density; // 데이타 영역의 밀도 계산
  if (density ≥ τ ) then
   Found = 1;
   Determine min, max for dense region;
   Update min, max of dense_region[];
  else if (density < τ ) then
   Found = 0;
  else  Found = null;
  return Found;  }
```

그림 3 밀집영역 결정 알고리즘

3.3 CLIP 알고리즘

3.3.1 점진적 프로젝션에 의한 클러스터링

CLIP에 의한 클러스터링 알고리즘은 그림 6과 같다. 데이타 공간의 점들은 데이타베이스에서 각각의 레코드를 의미한다. 이 때 입력 레코드의 각 애트리뷰트는 데이타 공간에서 하나의 차원을 이루게 되는데, 하나의 애트리뷰트에 해당하는 한 차원에 대하여 축-평행하게 프로젝션한다. 차원을 이루는 값들을 프로젝션한 후, 데이타의 분포를 계산하여 데이타의 밀도가 임계값 τ이상인 밀집영역을 찾아야 한다. CLIP은 데이타 집합의 밀도계산을 위해 3.2절에서 언급한 그리드-기반 접근방법의 원리를 사용한다. 그 다음 밀집영역에 해당하는 초월사각형(hyper-rectangle) 부분에 존재하는 레코드에 대해서만 그 다음 차원 값을 프로젝션한다. 부분차원에서 이러한 초월사각형 안의 데이타 점들의 밀도는 평균 밀도보다 매우 크다. 전체 데이타 공간을 k차원이라 할 때, 1차원에서 k차원에 이르기까지 순차적으로 각 차원에 해당하는 밀집영역을 그 다음 차원에 종속적으로 반영시킴으로써 최종적으로는 탐사할 데이타 공간 및 잡음을 줄인다. 그림 4와 같이, 각 차원에 있어서 이전 차원에 종속적으로 결정된 영역의 밀도가 기준이 되는 임계값을 초과하지 않는 차원에 대해서는 제외시키고 그 다음 차원을 조사한다. 왜냐하면 데이타가 비교적 균등하게 분포한 영역이거나 또는 임계값을 초과하지 않는 영역은 클러스터 형성에 관련성이 매우 적은 차원이므로, 이러한 차원에서는 영역을 선택하지 않는다.

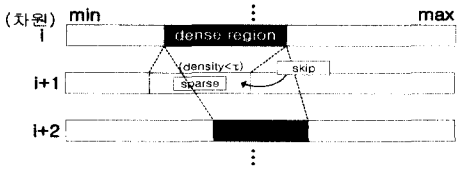

그림 4 밀집영역이 없는 경우

(차원)	1	2	3	4	···	k-m	k-m+1	···	k-1	k
1	1									
2	1	1								
3	0	1								
4	1	0		0						
⋮	⋮	⋮								
k-m	1					1				
k-m+1	⋮	⋮				⋮				
⋮	1					1	⋮			
k-1	0					0				
k	1									

그림 5 차원의 연관성 상태 배열

CLIP은 클러스터 형성에 연관된 부분차원을 선택하여 해당하는 부분차원에서 클러스터를 탐사하는 기법이다. 그러나 주어진 후보 차원의 조합에 대한 모든 경우의 수를 고려할 수 없으므로, 클러스터를 형성하는 차원의 최소 개수(Num_subspace)를 입력 값으로 정한다. 그 다음, 그림 5와 같은 배열 *state_list*를 초기화하는데 *state_list*는 클러스터를 형성하는 차원의 연관성을 나타낸다. *state_list*의 각 요소는 밀집영역에 관한 통계 정보를 저장하고 있으며, 각 요소에 관한 데이타 구조는 아래의 구조체 *state_list_element*와 같다. 프로젝션에 의하여 1차원에서 시작하여 k차원까지의 밀집영역의 조사가 끝나면, 1차원에서 선택되지 않은 데이타 점들의 2차원 영역을 검사하여 위의 과정을 k차원까지 반복한다. 이는 1차원을 제외한 나머지 차원들로 형성된 클러스터를 찾기 위한 목적이다. 이러한 과정은 (k-Num_subspace+1)차원에서 시작하는 클러스터 탐사를 끝으로 종료한다. 그 다음은 클러스터 형성에 관련된 차원에 대해 해당하는 영역을 보다 정확히 명세하기 위해 백 트레이스(back trace) 과정을 수행한다. 최종적으로 결정된 데이타 집합에 대해 백 트레이스의 수행이 끝나면 관련된 차원 및 해당하는 영역의 정확한 최소값, 최대값과 함께 하이퍼큐브(hypercube) 형태로써 클러스터에 대한 각 차원의 명세를 얻을 수 있다.

```
struct state_list_element {
        boolean flag; /*밀집영역의 존재 여부*/
            int count; /*객체의 개수*/
            double average; /*객체들의 평균 값*/
            double min; /*영역의 최소 값*/
            double max; /*영역의 최대 값*/
};
```

```
/* Input: Data_rec(입력 데이터 레코드 집합), ξ(그리드 간격의 크기)
        Num_subspace(클러스터로 인정할 부분차원 개수)
   Output: Identified Cluster(명세된 클러스터), 데이타 ID */
procedure GenerateCluster(Data_rec, Num_subspace) {
 Initialize the state_list[k][k], dense_region[];
 Sort the attributes of k-dimensions by descending order;
 for k-dimensions {
   for i=1 to (k-Num_subspace+1) do {
      Get the Object IDs of i-dimension except for the previously examined
      IDs;
      Found = Find_Dense(Data_rec);
      if (Found equal to 0) then
         i = i + 1;
      for j=i+1 to k do {
         Get the Object IDs of (i+1)-dimension dependent on i-dimension;
         Found = Find_Dense();
        Determine the each value(0, 1, null) of state_list[i][j] using Found;
         if (j equal to (k-Num_subspace+2)) then
           Check the number of "1" from the state_list;
           if the number is 1 then
             break For (i+1≤j≤k);
         else j = j + 1;
      }end For j=i+1 to k
      i = i + 1;
   }end For i=1 to (k-Num_subspace+1)
   Read object IDs within min, max in dense_region[];
   Describe min, max for each dimension in the object IDs using BackTrace
;
 }end For (k-dimensions)
}end Procedure GenerateCluster
```

그림 6 CLIP의 클러스터링 알고리즘

3.3.2 백 트레이스에 의한 클러스터 명세

그림 7은 CLIP에 의한 점진적인 프로젝션으로 클러스터링을 하는 예이다. 실험 데이타는 전체 5차원의 100,000개 데이타로서, 값의 범위는 [0,100], 그리드 간격 ξ는 10이며 클러스터로 인정할 부분차원의 개수(Num_subspace)는 3이다. 1차원에서 Reg_ID가 ⑤번 및 ⑨번인 영역을 시작으로 CLIP 알고리즘을 수행한 결과, 조사된 클러스터의 개수는 3개이며 Cluster ID는 각각 A, B, C이다.

그림 8은 그림 7에서 조사된 클러스터의 생성과정을 연결리스트 형태로 보이고 있다. 점진적인 프로젝션에 의한 최종적인 데이타 집합을 클러스터로 결정한 후, 그

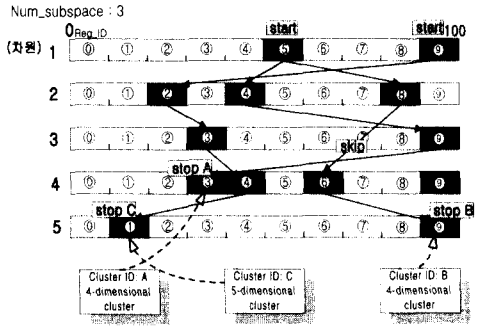

그림 7 CLIP에 의한 클러스터링 예

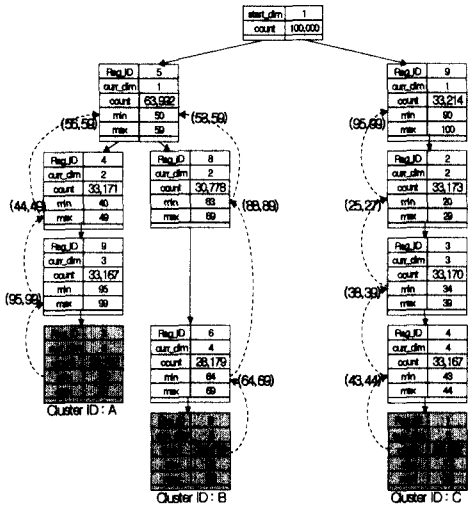

그림 8 백 트레이스에 의한 클러스터 명세

림 8에서 점선으로 표시한 것처럼 백 트레이스를 수행하면서 클러스터의 각 차원에 대한 최소(min), 최대(max) 값을 명세한다. 각 클러스터의 특징은 다음과 같다.

클러스터 형성에 5개 차원 모두가 관련된 전체차원 클러스터는 1개이며 Cluster ID가 C인 것으로 조사되었다. 또한 부분차원 클러스터는 2개로 조사되었는데 Cluster ID가 A와 B인 것으로서 클러스터 A는 1,2,3,4 차원으로 그리고 클러스터 B는 1,2,4,5차원으로 형성된다. 예를 들어, Cluster ID가 A인 클러스터에 대해 백 트레이스를 이용하여 각 차원을 명세하면, 데이타 개수는 33,167이며, 클러스터를 형성하는 4차원 값의 범위는 [30, 31], 3차원은 [95, 99], 2차원은 [44, 49] 그리고 1차원은 [56, 59]이다. 클러스터 A의 1차원 값의 경우, 클러스터 결정 과정에서 조사된 영역의 범위는 [50, 59]인데 백 트레이스를 통해 명세된 범위는 [56, 59]로서 더욱 정확한 차원 명세를 얻을 수 있었다. 또한 점진적으로 차원이 증가함에 따라 데이타 수가 줄고 있는 현상을 볼 수 있는데, 이는 잡음 데이타를 필터링하고 있음을 의미한다.

4. 실험 및 성능 평가

CLIP의 성능을 평가하기 위해 [4]에서 사용한 벤치마킹된 합성 데이타 집합을 이용하여 실험하였으며, 실험의 목표는 CLIP의 효율성과 효과성을 실험적으로 평가하는 데 있다. 효율성은 데이타 공간의 차원, 클러스터의 차원, 데이타베이스의 크기에 따라 수행시간이 어떻게 증가하는가를 증명하는 것이다. 효과성은 클러스터링 결과의 정확성을 의미하는데 CLIP이 고차원 데이타 공간의 임의 부분차원에서 클러스터를 발견할 수 있는가를 실험하는 것이다.

실험 환경은 256M 메인 메모리의 200-MHz SUN-Ultra SPARCⅡ워크스테이션에서, 데이타는 12GB SCSI 디스크에 저장하였다. 또한 C++언어와 LEDA- 4.2 라이브러리 그리고 GNU g++ 컴파일러를 사용하여 구현하였다.

4.1 부분차원 알고리즘의 비교평가

본 논문에서 제안하는 CLIP의 성능 및 장·단점을 기존의 부분차원 클러스터링 기법과 비교하였다. [표 1]에서 볼 수 있듯이 CLIP의 성능은 데이타베이스에서 스캔하는 객체의 수가 CLIQUE나 PROCLUS보다 적음을 알 수 있으므로 알고리즘의 효율성 측면에서 우수하다고 평가할 수 있다. CLIP의 수행성능에 대한 연산은 아래와 같이 증명할 수 있다. 단, [표 1]에서 제시한 CLIQUE와 PROCLUS의 수행 성능은 각각 [2],[4]에서 발표한 결과를 참조한 것이다.

[증명]

k차원에서 N개의 데이타에 대해 m개 차원 이상으로 형성된 클러스터의 탐사를 위해 데이타를 액세스하는 총 횟수 $a(k,m)$는 다음과 같이 정의할 수 있다. 여기서, $c^{m-2}N$는 $(m-1)$개 차원으로 형성된 클러스터에서 액세스되는 데이타 수이다.

$$a(k,m) \le a(k-1,m-1) + c^{m-2}N$$
$$a(k,m) \le a(k-2,m-2) + c^{m-3}N + c^{m-2}N$$
$$\vdots$$
$$a(k,m) \le a(k-m+1,1) + c^{0}N + \cdots + c^{m-3}N + c^{m-2}N$$
$$a(k,m) \le (k-m+1)N + N + cN + \cdots + c^{m-2}N$$
$$a(k,m) \le (k-m+1)N + \frac{1-c^{m-1}}{1-c}N$$

표 1 부분차원 클러스터링 기법의 장·단점 및 수행성능

클러스터링 / 알고리즘	장 점	단 점	수행 성능
CLIQUE	· 입력 데이타 순서와 무관한 동일한 결과 · 고밀도 클러스터를 갖는 부분공간 자동 식별	· 그리드 셀의 지수적인 증가 · 많은 부분공간이 제거될 가능성 · 클러스터간에 큰 오버랩이 존재할 가능성	$O(c^k + N \cdot k)$ k : 차원 수 N : 입력 데이타 수 c : 상수 ($c \geq 1$)
PROCLUS	· 클러스터 형성에 연관성이 낮은 차원의 제거로 데이타의 잡음 감소 : 향상된 효율성	· 고차원 공간에서 특정 차원에 국한된 희소 데이타 분석에만 용이 · 고차원 공간에서 최상 medoid 선택의 어려움	$O(N \cdot k \cdot m)$ N : 입력 데이타 수 m : 클러스터 수 k : 차원 수
CLIP	· 고차원 데이타의 선형적 처리로 연산비용 감소 · 점진적 프로젝션으로 조사 공간 및 잡음 감소	· 그리드 접근방식의 근본적인 문제점 : 클러스터 품질이 그리드 셀의 크기, 기준 밀도에 다소 의존적일 가능성	$O(N \cdot k \cdot c)$ N : 입력 데이타 수 k : 차원 수 c : 상수($0 < c \leq 1$)

$0 \leq c < 1$이므로, $\frac{1-c^{m-1}}{1-c} \leq (m-1)$

따라서, $\frac{1-c^{m-1}}{1-c} = (m-1)$라 가정할 경우

$a(k, m) \leq (k-m+1)N + (m-1)N = k \cdot N$ 이다.

4.2 CLIP의 실험 결과

본 논문에서는 특정한 부분차원에 고밀도 클러스터를 포함하는 데이타 집합을 생성하기 위해 [4]의 합성 데이타 생성기를 이용하였다. 실험에서 값의 범위는 모든 애트리뷰트에 대해 [0,100]으로 고정하였으며, 그리드 셀의 크기인 ξ는 10으로 수행하였다. 모든 시간은 초 단위이다.

4.2.1 데이타베이스 크기

그림 9는 100,000에서 500,000 레코드까지 데이타베이스의 크기를 증가시킴에 따른 확장성을 보여준다. 데이타 공간은 50차원이며 5개의 클러스터가 있다. 각각 서로 다른 5차원 부분공간에서 선택률 τ는 10%로 하였다. 예상대로 수행시간은 데이타베이스 크기에 선형적으로 증가하였는데, 이는 데이타베이스를 스캔하는 횟수는 변하지 않기 때문이다.

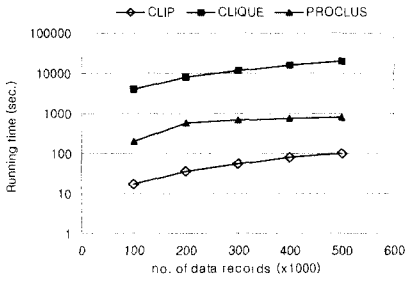

그림 9 데이타 개수에 대한 클러스터링 시간

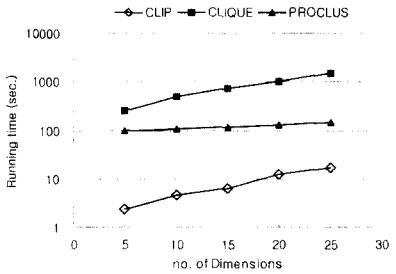

그림 10 데이타 차원에 대한 클러스터링 시간

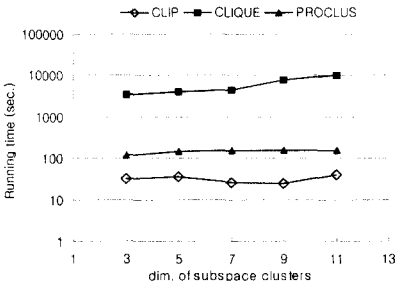

그림 11 숨겨진 클러스터의 탐사 시간

4.2.2 데이타 공간의 차원

그림 10은 데이타 공간의 차원을 5에서 25까지 증가시킴에 따른 확장성을 보이고 있다. 데이타베이스는 100,000 개의 레코드를 가지며 5개 클러스터가 있다. 각각 서로 다른 5차원 부분공간에서 τ는 10%로 정한다. 흥미있는 부분차원을 찾는 문제는 데이타 공간의 차원이 증가하여도 수행시간은 잘 확장하지 않는다는 것에

주목할 수 있다.

4.2.3 숨겨진 부분차원 클러스터의 차원

그림 11은 전체 25차원 공간에서 숨겨진 클러스터의 차원을 3에서 11로 증가시킴에 따른 확장성을 보여주고 있다. 각 경우에서, 데이타베이스는 100,000 개의 레코드를 가지며 1개의 클러스터가 최고차원의 적당한 부분공간에 포함된 상태이다.

4.2.4 정확도

이상의 모든 실험에 있어서, 원래의 클러스터들은 CLIP 알고리즘에 의해 모두 조사되었다. 일부의 경우에서는, 매우 작은 선택률을 가진 단일 밀집단위로 형성된 소수의 추가 클러스터들이 조사되었다. 그러나 이러한 인공물은 데이타 생성 알고리즘의 부산물이며 τ가 매우 낮다는 사실을 발견할 수 있었다.

4.3 BIRCH, DBSCAN과의 성능비교

4.2절에서 사용한 [4]의 데이타 집합으로 CLIP을 BIRCH, DBSCAN과 비교하여 실험하였다. 이 실험의 목적은 BIRCH, DBSCAN과 같은 전체 차원 기반의 클러스터링 알고리즘이 부분차원 클러스터링에서 사용할 수 있는가를 평가하는 것이다. 이러한 알고리즘들의 설계 목표인 전체 차원공간에서 클러스터를 발견하는 태스크에 있어서는 CLIP이 특별한 이점을 갖지 않는다. 5에서 10까지 차원을 다양하게 하면서 5차원 부분공간에 포함된 클러스터를 사용하였다. 참고로, CLIP은 모든 경우에서 클러스터를 모두 조사할 수 있었다. 단, [표 2]는 [4]에서 제시한 실험결과를 참조한 것이다.

BIRCH의 입력으로는, 100,000 개의 데이타 점으로 구성된 집합에 대해 5개의 클러스터를 제공하였다. 입력 클러스터들은 5차원 부분공간에서 초월사각형에 해당하며, 남은 애트리뷰트의 값들은 균등하게 분포되었다. [표 2]에서 요약된 결과는 BIRCH가 10차원 공간에 숨겨된 5차원 클러스터들을 찾아내는 것을 보인다. 그러나 데이타 공간의 차원이 증가할 경우 실험은 실패하였다. 이것은 BIRCH가 모든 차원을 고려하는 거리 함수를 사용하기 때문이라고 분석된다. 데이타 공간의 차원이 증가함에 따라, BIRCH는 입력 매개변수를 5로 하여도 5개의 클러스터를 반환하지 못했다. 다른 데이타 집합에 대하여, 3, 4 또는 5개의 클러스터를 반환하였다.

DBSCAN은 자체에서 클러스터의 수를 발견하므로 클러스터의 개수를 입력 값으로 정하지 않아도 된다. DBSCAN은 데이타가 10차원 이상에서는 수행할 수 없었다. 입력은 10,000개의 데이타 집합으로 하였다. BIRCH의 실험과 같이, 클러스터들은 5차원 부분공간에 있다. [표 2]의 결과에서 볼 수 있듯이 DBSCAN은 10차원 데이타 공간에서 5차원 클러스터를 조사하지 못했다. 단, 공간의 차원을 7로 줄였을 때 찾을 수 있었다. 그리고 8차원 데이타 공간에서도 숨겨진 5차원 클러스터들 중 오직 하나만 조사할 수 있었다.

표 2 비교 실험 결과

데이타의 차원	클러스터의 차원	클러스터 수	찾은 클러스터 수			식별된 실제 클러스터 수		
			BIRCH	DBSCAN	CLIP	BIRCH	DBSCAN	CLIP
5	5	5	5	5	5	5	5	5
10	5	5	5	1	5	5	0	5
20	5	5	3,4,5	-	5	0	-	5
30	5	5	3,4	-	5	0	-	5

5. 결 론

본 논문에서는 고차원 데이타의 클러스터링을 위한 CLIP 알고리즘을 제안하였다. CLIP은 고차원 데이타 공간의 부분차원에서 흥미있는 패턴을 발견하기 위해 점진적인 프로젝션을 이용하는 클러스터링 기법이다. 이와 같은 부분차원 클러스터링은 입력된 데이타 점들의 다양한 부분집합간에 존재하는 상관관계를 다룰 수 있다는 장점이 있다. CLIP의 주요 사상인 프로젝션 방법은 고차원에 대한 선형변환이며, 이는 고차원에서 급격하게 성능이 저하되는 "차원의 저주" 현상을 극복하기 위해 사용되었다. 또한 CLIP은 프로젝션을 점진적으로 적용하여 잡음 데이타를 현저하게 감소시키는 효과를 제공하였다.

실험적인 평가로는 CLIP의 수행시간이 입력 크기에 대해 선형적으로 증가하는 확장성을 보이며, 데이타의 차원의 크기나 클러스터를 포함하고 있는 최고 차원을 증가시키는데 있어서도 좋은 확장성을 갖는다. CLIP은 전체 데이타 공간에 클러스터가 존재하지 않더라도, 저차원 부분공간에 숨겨진 클러스터를 정확하게 발견할 수 있었다. CLIP에 의한 자동화된 부분공간 클러스터링의 계산적인 실행가능성을 평가함으로써, 분류, 시계열 클러스터링, 순차패턴 탐사, 연관규칙 같은 다른 기법과 더불어 기본적인 데이타 마이닝으로 고려될 수 있을 것이다. 자동화된 부분공간 클러스터링은 데이타 마이닝 이외의 분야에서도 유용하게 사용될 수 있다. 예를 들어, OLAP 데이타 큐브에 있어서 밀집 및 희소한 데이타 영역에 대한 차원의 명세나 영역질의를 위한 차원의 명세가 요구되는데 CLIP은 이와 같은 목적에도 사용될 수 있다. 현재 CLIP의 응용분야로서 전자상거래에서 발생되는 방대한 양의 데이타를 클러스터링하여 판매전략, 수요예측, 생산계획

등에 적용할 수 있는 연구가 진행되고 있다.

향후 연구는 클러스터의 품질을 향상시키는 방안에 관한 것으로서, 클러스터의 결정에 중요한 영향을 미치는 밀도 임계값이나 그리드의 크기 등 입력 매개변수의 선택을 지원하는 시스템을 개발할 계획이다.

참 고 문 헌

[1] Mihael Ankerst, Markus M. Breunig, Han-Peter Kriegel, and Jorg Sander, "OPTICS: Ordering points to identify the clustering structure," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, 1999.

[2] Charu C. Aggrawal, Cecilia Procopiuc, Joel L. Wolf, Philip S. Yu, and Jong Soo Prk, "Fast Algorithms for Projected Clustering," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 61-72, 1999.

[3] S. Berchtold, D. A. Keim, C. Böhm, H.-P. Kriegel, "A Cost Model For Nearest Neighbor Search in High-Dimensional Data Space," *Proc. of the 16th Symposium on Principles of Database Systems (PODS)*, pp. 78-86, 1997.

[4] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan, "Automatic subspace Clustering on High Dimensional Data Mining Applications," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 94-105, 1998.

[5] Raymond T. Ng, Jiawei Han, "Efficient and Effective Clustering Methods for Spatial Data Mining," *Proc. of 20th Int. Conf. on VLDB*, pp. 144-155, 1994.

[6] Martin Ester, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu, "A density-based algorithm for discovering clusters in large spatial database with noise," *Proc. of Int. Conf. on Knowledge Discovery and Data Mining*, 1996.

[7] Hinneburg A., Keim D. A, "An Efficient Approach to Clustering in Large Multimedia Databases with Noise," *Proc. of 4th Int. Conf. on Knowledge Discovery and Data Mining*, 1998.

[8] Tian Zhang, Raghu Ramakrishnan, and Miron Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pp. 103-114, 1996.

[9] Wei Wang, Jiong Yang, and Richard Muntz, "STING: A Statistical Information Grid Approach to Spatial Data Mining," *Proc. of 23rd Int. Conf. on VLDB*, pp. 186-195, 1997.

[10] Christos Faloutsos, "Fast Searching by Content in Multimedia Database," *Data Engineering Bulletin*, 18(4), 1995.

[11] Fayyad, U. M., et al., *Advances in Knowlwdge Discovery and Data Mining*, AAAI Press / The MIT Press, pp. 307-328, 1996.

[12] Kaushik Chakrabarti, Sharad Mehrotra, "Local Dimensionality Reduction: A New Approach to Indexing High Dimensional Spaces," *Proc. of 26th Int. Conf. on VLDB*, pp. 89-100, 2000.

[13] Hinneburg A., "Mining for High Dimensional Cluster using Projection and Visualizations," *Proc. of the EDBT 2000 phD Workshop*, 2000.

[14] Hinneburg A., Keim D. A, "Opimal Grid-Clustering: Towards breaking the Curse of Dimensionality in High-Dimensional Clustering," *Proc. of 25th Int. Conf. on VLDB*, pp. 506-517, 1999.

[15] 이혜명, 박영배, "고차원 데이타에서 점진적 프로젝션을 이용한 클러스터링", 한국정보과학회 가을학술발표논문집(I), 2000.

이 혜 명

1989년 2월 명지대학교 공학사(전자계산학). 1993년 2월 명지대학교 공학석사(전자계산학). 1997년 2월 명지대학교 박사과정 수료(컴퓨터공학). 1998년 3월 ~ 현재 경문대학 컴퓨터정보과 조교수. 관심분야는 데이타마이닝, 웹 DB, 전자상거래 등

박 영 배

1974년 2월 동아대학교(공학사) 전기공학. 1980년 2월 연세대학교(공학석사) 전자계산학. 1993년 2월 서울대학교(공학박사) 컴퓨터공학. 1974년 3월 ~ 1981년 2월 한국전력공사 전자계산소 과장대리. 1990년 3월 ~ 1992년 2월 명지대학교 전자계산소 소장. 1993년 ~ 2000년 중앙전산개발경진대회(행정자치부) 심사위원장. 1997년 3월 ~ 2001년 8월 산업대학원 원장. 1981년 3월 ~ 현재 명지대학교 컴퓨터공학과 교수 재직중이며, 데이타베이스, 자료구조, 파일처리 등을 강의하고 관심분야로는 1. Spatial, Multidimensional, Web, Large fingerprint Databases, 2. Data Warehousing and Data Mining, 3. System Integration 등