Journal of Digital Convergence
Vol. 20. No. 2, pp. 217-223, 2022

ISSN 2713-6434 / eISSN 2713-6442
https://doi.org/10.14400/JDC.2022.20.2.217

온라인 데이터 스트림에서의 동적 부분 공간 클러스터링 기법

박남훈
안양대학교 융합소프트웨어학과 교수

Dynamic Subspace Clustering for Online Data Streams

Nam Hun Park
Professor, Dept. of Convergence Software, Anyang University

요 약 온라인 데이터 스트림에 대한 부분 공간 클러스터링은 데이터 공간 차원의 모든 부분 집합을 검사해야 하므로 많은 양의 메모리 자원을 필요로 한다. 유한한 메모리 공간에서 데이터 스트림에 대한 클러스터들의 지속적인 변화를 추적하기 위해 본 논문에서는 메모리 자원을 효과적으로 사용하는 격자기반 부분 공간 클러스터링 알고리즘을 제안한다. n차원 데이터 스트림이 주어지면 각 차원 데이터 공간에 있는 데이터 항목의 분포 정보를 격자셀 리스트에 의해 모니터링 된다. 첫번째 레벨의 격자셀 목록에서 데이터 항목의 빈도가 높아 단위 격자셀이 되면 해당 격자셀로부터 모든 가능한 부분 공간의 클러스터를 찾기 위해 다음 레벨의 격자셀 리스트를 자식 노드로 생성한다. 이와 같이 최대 다차원 n레벨의 격자셀 부분 공간 트리가 구성되고, k차원의 부분 공간 클러스터는 부분 공간 격자셀 트리의 k레벨에서 찾을 수 있다. 실험을 통해서 제안하는 방법이 기존 방법만큼 정확도를 유지하면서, 밀집 공간만 확장하여 컴퓨팅 자원을 보다 효율적으로 사용하는 것을 확인하였다.

주제어 : 데이터 스트림, 클러스터링, 데이터 마이닝, 부분 공간 클러스터링, 온라인 데이터 마이닝

Abstract Subspace clustering for online data streams requires a large amount of memory resources as all subsets of data dimensions must be examined. In order to track the continuous change of clusters for a data stream in a finite memory space, in this paper, we propose a grid-based subspace clustering algorithm that effectively uses memory resources. Given an n-dimensional data stream, the distribution information of data items in data space is monitored by a grid-cell list. When the frequency of data items in the grid-cell list of the first level is high and it becomes a unit grid-cell, the grid-cell list of the next level is created as a child node in order to find clusters of all possible subspaces from the grid-cell. In this way, a maximum n-level grid-cell subspace tree is constructed, and a k-dimensional subspace cluster can be found at the k^{th} level of the subspace grid-cell tree. Through experiments, it was confirmed that the proposed method uses computing resources more efficiently by expanding only the dense space while maintaining the same accuracy as the existing method.

Key Words : Data Streams, Clustering, Data mining, Subspace clustering, Online data mining

*This paper was supported by Anyang University Research Grant.
*Corresponding Author : Nam Hun Park(nmhnpark@anyang.ac.kr)
Received November 16, 2021 Revised December 13, 2021
Accepted February 20, 2022 Published February 28, 2022

1. 서론

최근 인터넷 스트림, 실시간 센서 데이터 등 실시간 데이터 스트림 환경이 증가함에 따라 지속적으로 생성되는 방대한 데이터 스트림에 대한 연구가 많은 관심을 받아왔다. 데이터 스트림은 빠른 속도로 지속적으로 생성되는 방대한 데이터로 데이터 스트림의 모든 항목을 유지 관리하는 것은 불가능하다[1,2]. 따라서 데이터 스트림을 처리하기 위해서는 다음 사항[3-5]을 충족해야 한다. 첫째, 데이터 스트림 내의 각 데이터 항목은 반복없이 최대한 번만 탐색되어야 한다. 둘째, 데이터 스트림의 새로운 데이터 항목은 연속적으로 계속 생성되는데 비해 이를 처리하는 메모리 자원의 용량은 유한하게 제한되어 있다. 셋째, 새로 생성된 데이터 항목은 최대한 빠르게 처리하여 데이터 스트림의 분석 결과가 즉시 활용될 수 있도록 해야 한다.

클러스터링은 데이터 집합을 클러스터라고 하는 의미있는 집단으로 구분하는 데이터 마이닝 기법으로 마케팅 전략을 위한 고객 군집, 선거전략을 위한 유권자 군집 분석, 상품 추천을 위한 소비자 분석 등 다양한 분야에 활용되고 있으나, 분석 알고리즘의 복잡성으로 스트림 데이터에 대해서는 쉽게 적용하지 못하고 있다. 데이터 스트림에 대해 일부 클러스터링 알고리즘이 연구되었지만 데이터 스트림에서 대한 부분 공간 클러스터링 알고리즘은 수행과정에서 복잡도로 인해 아직 제시되지 않았다.

실제 응용 데이터는 속성이 많은 고차원 데이터로 다차원 클러스터보다 부분 공간에서 클러스터를 추출하는 것이 더 바람직하다[6,7]. 고차원 데이터 세트의 경우 적합한 다차원 클러스터가 없거나 이러한 클러스터를 찾는 것이 사용자에게 거의 유용하지 않을 수 있다. 반면에, 일부 차원의 부분 공간 클러스터는 관련된 차원 간에 중요한 정보를 제공할 수 있다. n차원 데이터 공간 $\boldsymbol{N}=N_1xN_2x\cdots xN_n$이 주어지면 N차원의 일보 차원으로 구성된 데이터 공간 N'을 데이터 공간 $\boldsymbol{N}$의 부분 공간이라고 한다. 예를 들어, 3차원 데이터 공간 $\boldsymbol{N}=N_1xN_2xN_3$이 주어지면 의미 있는 클러스터는 N_1 및 N_2에 의해 형성된 2차원 부분 공간 내부에 존재할 수 있다. 데이터 공간 $\boldsymbol{N}$의 k-차원 부분 공간은 k개의 선택된 차원 세트에 의해 형성된 직사각형 데이터 공간에 의해 정의된다. 데이터 공간의 차원이 클수록 이러한 부분 공간 클러스터를 추출하는 것이 더 바람직하며, 실제 일부 누락된 차원 값이 있는 데이터 항목은 부분 공간 클러스터링을 통해 효율적으로 처리할 수 있다[8].

기존의 셀트리[9]에서는 데이터 스트림에 대한 군집분석 방법을 제시하였으나, 응용환경에서 실제 데이터를 대상으로 데이터 차원이 클수록 기존 다차원 클러스터링 방법은 부분공간 클러스터링에 비해 좋은 분석 결과를 찾지 못하였다. 본 논문에서는 데이터 스트림에 대한 부분 공간 클러스터링을 위한 방법을 제시한다. 온라인 데이터 스트림을 통한 부분 공간 클러스터링은 차원의 모든 부분 집합을 검사해야 하므로 많은 양의 메모리 공간이 필요하다. 이를 극복하기 위해 메모리를 효율적으로 활용하는 방법을 사용한다. 격자 리스트의 전체 공간을 동시에 확장하지 않고 격자셀의 지지도를 비교하여 상대적으로 부분 공간 클러스터가 존재할만한 격자셀을 먼저 분할하여 각 부분 공간의 격자셀을 번갈아 가며 동적으로 확장한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 다차원 데이터 스트림에서 하위 클러스터를 찾기 위한 부분 공간 트리를 제공한다. 3장에서는 제안한 방법의 성능을 평가하기 위해 다양한 실험 결과를 비교 분석한다. 마지막으로 4장에서는 결론을 제시한다.

2. 관련 연구

데이터 스트림 클러스터링 방법으로 새로 생성된 데이터 항목을 각 부분 데이터 스트림으로 나누어 수행하는 *LSEARCH*[10] 기법이 있다. *O(1) K-medoid* 알고리즘을 사용하여 각 부분 데이터 스트림들에서 각각 k개의 클러스터 중심을 구한다. [11]에서는 모든 부분 데이터 스트림의 k개 클러스터 중심들에 대해 다시 클러스터링 과정을 반복해서 전체 데이터 스트림의 클러스터 중심들을 구한다. 하지만, 적절한 클러스터의 수 k를 쉽게 정의하지 못하는 경우, *LSEARCH*는 클러스터의 품질이 최대가 될 때까지 반복해서 수행하기 때문에 부하가 크게 증가한다. *VFKM*[12] 알고리즘은 *k-means* 알고리즘을 확장한 것으로 데이터 스트림을 대상으로 적절한 수의 클러스터를 찾도록 개선하였다. *Hoeffding* 경계를 적용하여 *k-means* 알고리즘의 각 단계에서 적절한 클러스터 수를 결정한다.

CluStream[13]은 데이터 스트림에서 클러스터 변화를 탐색하기 위해 제안되었다. 이 기법은 *k-means* 알고리즘을 수행하여 극소 클러스터들을 생성한다. 클러스터의 그룹 구조를 가능한 정확히 발견하기 위해 극소 클러스터의 수는 메모리 공간에서 사용가능한 최대의 수로

정해진다. 오프라인 과정에서 *CluStream*은 극소 클러스터 결과들이 누적된 집합에 대해 *k−means* 알고리즘을 수행하여 최종 *k*개의 최종 매크로 클러스터를 찾는다. 그러나 이 알고리즘은 오프라인으로 매크로 클러스터를 찾는 방법으로 온라인 데이터 스트림의 실시간 클러스터를 발견하는 데는 적합하지 않다. [14, 15]에서는 부분공간 클러스터링 방법을 제시하였다. *n*차원 데이터 공간에 대해서 2^n-1개의 부분공간을 생성하여 각각의 공간에서 클러스터를 탐색하여 많은 시간과 공간을 소모한다.

분 논문에서는 셀트리 기반의 부분공간 격자기반 클러스터링 방법을 제시한다. 격자셀 리스트로 데이터 공간을 특정 차원을 기준으로 분할하고, 각 노드에서 해당 격자셀의 데이터 객체 통계정보를 관리한다. 객체의 빈도가 높아 조밀한 격자셀은 정확한 클러스터의 범위를 찾기 위해 주어진 수의 크기가 동일한 보다 작은 격자셀로 분할한다. 객체 빈도가 높은 격자셀은 반복해서 최소크기 λ가 될 때까지 격자셀을 분할한다. 최소크기 격자셀은 자식노드로 다른 차원에 대한 격자셀 리스트를 생성한다. 이렇게 생성된 셀트리는 부모노드에서 빈발범위와 함께 다차원 클러스터를 구성하게 된다. *n*차원 데이터 공간에 대해서 깊이가 *n*인 부분공간 트리가 구성되어 다차원 클러스터를 찾을 수 있다.

3. 부분 공간 클러스터링

다차원 데이터 공간 $\boldsymbol{N}=N_1\times\ldots\times N_n$,에 대한 데이터 스트림은 다음과 같이 연속적으로 생성되는 데이터 항목의 무한 집합으로 다음과 같다.

i) t번째 턴에서 생성된 데이터 항목은 $e^t=<e_1^t,e_2^t,\cdots,e_d^t>$, $e_i^t\in N_i$, $1\le i\le d$로 표시된다.

ii) 현재 데이터 스트림 $\boldsymbol{D^t}$는 지금까지 생성된 모든 데이터 항목을 나타낸다. $\boldsymbol{D^t}=\{\boldsymbol{e^1},\boldsymbol{e^2},\cdots,\boldsymbol{e^t}\}$.

iii) 현재 데이터 스트림 $\boldsymbol{D^t}$에서 생성된 데이터 항목의 총 수는 $|\boldsymbol{D^t}|$는 수식(1)과 같이 이전 시간 t−1에서의 총수에 현재 개수를 더한 것으로 구한다.

$$|\boldsymbol{D^t}| = \begin{cases} 1 & \text{if } t=1 \\ |D|^{t-1}+1 & \text{if } t\ge 2 \end{cases} \quad (1)$$

현재 데이터 스트림 Dt에 대한 1차원 데이터 공간의 격자 셀은 정의 1에 정의되어 있다.

정의 1. 격자셀 $g(I,c,\mu,\sigma)$의 분포 통계

데이터 공간 내의 차원 N에서의 현재 데이터 스트림 $\boldsymbol{D^t}$에 대해 격자셀 $g(I, ct, \mu t, \sigma t)$는 범위 $I=[s,f)$ 내의 데이터 항목들의 분포 정보를 저장한다. 데이터 공간 N의 범위에서. $\boldsymbol{D_g^t}$는 격자셀 g의 범위에 있는 $\boldsymbol{D^t}$의 항목을 나타낸다. $e_i\in\boldsymbol{D^t}$ 및 $e_i\in I$ }. 격자셀 g 내의 데이터 분포 정보는 다음과 같다.

i) c^t : $\boldsymbol{D_g^t}$의 데이터 항목 수.

ii) μ^t : μ^t는 $\boldsymbol{D_g^t}$의 데이터 항목들의 평균

$$\mu^t = \sum_{j=1}^{c^t} e^j / c^t \quad (2)$$

iii) σ^t : σ_i^t는 $\boldsymbol{D_g^t}$의 데이터 항목들의 표준 편차

$$\sigma^t = \sqrt{\sum_{j=i}^{c^t} (e^j-\mu^t)^2 / c^t} \quad (3)$$

미리 정의된 분할 계수 h가 주어지면 1차원 데이터 공간 N의 범위는 h개의 동일한 크기의 격자셀 $G=\{g_1,g_2,\cdots,g_h\}$ 로 분할된다. 격자셀 g는 간격 $g.I=[I.s, I.f)$에 있는 데이터 항목의 분포 정보를 수식(2), 수식(3)과 같이 평균과 편차로 관리한다. 격자셀 g의 지지도는 $\boldsymbol{D^t}$의 총 데이터 항목 수에 대한 격자 셀 간격 내부에 있는 데이터 항목 수의 비율이다($g.c^t/|\boldsymbol{D^t}|$). 데이터 항목의 빈도가 잦아 격자셀의 지지도가 주어진 S_{par}보다 크거나 같을 때, 해당 격자셀은 h개의 동일한 크기의 격자셀로 분할된다. 이러한 분할은 데이터 공간의 밀집된 영역에서 반복적으로 수행되어 각 격자셀의 간격은 해당 격자셀의 데이터 빈도에 따라 다르게 된다. 데이터 공간의 전체 범위에서 동적으로 변화하는 격자셀을 효율적으로 관리하는 격자셀 리스트는 다음과 같이 정의된다.

정의 2. 격자셀 리스트 $\boldsymbol{S}$

각 차원 데이터 공간 N의 현재 데이터 스트림 $\boldsymbol{D^t}$가 주어지면 차수가 m인 형제 목록 $\boldsymbol{S}$는 다음과 같이 정의된다.

i) 격자셀 목록 $\boldsymbol{S}=<E_1,E_2,E_3,\cdots,E_p>$는 격자셀 엔트리 $E_1,E_2,\cdots,E_p$의 단일 연결 목록

ii) 각 형제 항목 E_i는 데이터 구조 $E(min,max,G[1,\cdots,m],next_ptr)$를 유지

iii) $G[1,\cdots,m]$의 m 슬롯은 최대 m 격자 셀을 보유

iv) $v(<m)$ 슬롯 이 비어 있지 않은 경우 항목 E_i의 범위는 $range(E_i)=\bigcup_{j=1}^{v} G[j].I$ 로 정의되고 $[min,max)$로 표시

v) $next\_ptr$: S의 다음 격자셀 목록에 대한 포인터.

S의 범위는 S의 수식 (4)와 같이 모든 형제 노드 항목 범위의 합집합으로 정의되며 데이터 공간 N의 전체 범위와 동일하다.

$$\text{range}(S) \equiv \bigcup_{i=1}^{p} range(E_i) \equiv range(N) \quad (4)$$

첫 번째 격자셀 엔트리로부터 생성되는 S의 모든 형제 항목에는 주어진 분할 계수 h에 대해 항상 개의 최소한 $\lceil (m-h+2)/2 \rceil$ 개의 격자 셀이 있다.

n차원 데이터 공간 N의 각 차원 N_1, N_2,…,N_n 상에서 k차원 격자셀의 직사각형 부분 공간은 $RS=I_1xI_2x\cdots I_k$ $(1\le k\le n)$로 정의된다. 이러한 격자셀의 부분 공간에서 데이터 항목의 분포 정보를 효율적으로 모니터링하기 위해 정의 3에 정의된 m차 부분 공간 트리가 사용된다. multi-way tree의 첫 번째 자식-형제 표현에 따라 부분 공간 트리의 노드가 구성된다.

정의 3. 부분 공간 트리의 구조

$N_1\rightarrow N_2\rightarrow\cdots\rightarrow N_n$ 차원 순서의 n차원 데이터 공간 $N=N_1xN_2x\cdots xN_n$에서의 부분 공간트리는 다음과 같다.

i) 부분 공간 트리의 각 노드는 격자셀 리스트 S와 자식 노드 배열 $U[1,\cdots m][1,\cdots,n-1]$을 가진다. m개의 격자셀은 최대 n-1 레벨의 자식 노드를 가진다.

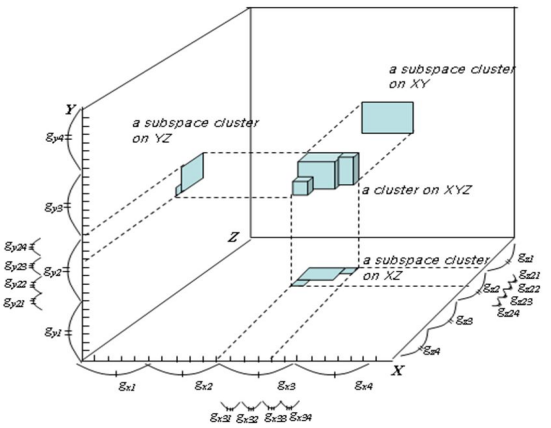
ii) T_{dim} : 격자셀 리스트가 정의되는 차원 k

iii) 트리의 f^{th} 레벨의 노드 항목 p가 $(j-1)^{th}$ 레벨의 노드 항목 q의 격자 $q.G[i]$ $(1\le i\le m)$의 자식노드라면, 노드 p는 새로운 형제 목록 S의 첫 번째 형제 항목이 되고 노드 p는 형제 목록 S의 헤드 노드가 된다..

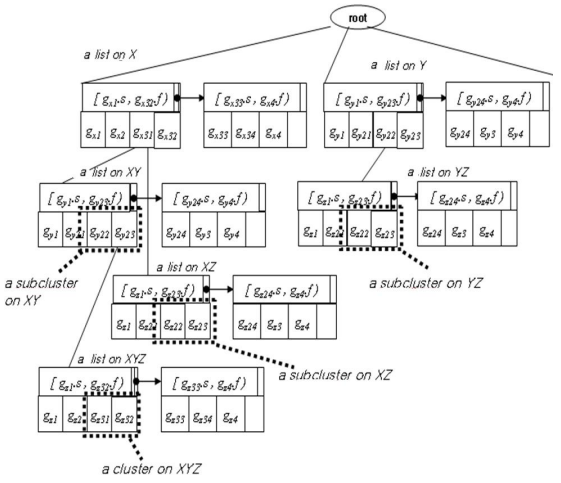
iv) 격자셀 $q.G[i]$는 형제 목록 S에 있는 모든 격자 셀의 부모 격자 셀로 호출된다.

v) D_g^t가 격자셀 g_j의 범위에 있는 데이터 항목을 나타내도록 하십시오. 즉, $D_g^t=\{\ e|\ e\in D^t$ 및 $e\in RS(g_j)\ \}$. 직사각형 부분 공간 $RS(g_j)$에서 데이터 항목의 분포 정보는 격자셀 $g_j(I,c,\mu,\sigma)$에 의해 모니터링된다. 즉, D_g^t의 현재 데이터 항목 수는 $g_j.c^t$에 의해 저장된다. 또한, D_g^t의 데이터 항목의 평균 및 표준 편차는 각각 $g_j.\mu$ 및 $g_j.\sigma$에 의해 저장된다.

차원 $N_l(v+1\le l\le n)$에 대한 격자셀 리스트의 격자셀이 빈도가 높아지면 h개의 더 작은 격자 셀로 분할된다. 결과적으로 격자셀 리스트의 격자셀 수가 증가한다. 또한, 분할을 반복하여 최소 단위 격자셀이 되면 다음 순서 차원에 대한 $(n-1)$개의 새 격자셀 리스트가 다음 레벨에서 빈도 높은 격자셀의 자식으로 생성된다. 이런 식으로 부분 공간 트리는 최대 n번째 수준까지 성장한다. 그림 1은 XYZ 데이터 공간에 있는 부분 공간 트리의 예를 보여준다. 한편, 부분 공간 트리에서 노드 n의 격자셀 g가 희소화되면 병합되고, 이에 속한 모든 자식 노드들도 가지치기 된다.



(a) A subspace cluster on $XxYxZ$
(b) A subspace tree
Fig. 1. A subspace tree for XYZ data space



정리 1. 분할 계수 h와 차원 시퀀스 $N_1\rightarrow N_2\rightarrow\cdots\rightarrow N_d$가 주어지면 d차원 데이터 공간의 데이터 스트림에 대해 $v^{th}(v\le d)$ 레벨의 격자셀 리스트의 수 $sl(v)$에 대해 범위 크기가 λ인 단위 격자셀 $g(I,c^t,\mu^t,\sigma^t)$을 생성하는 데 필요한 최소 분할 작업 $P_{min}(g)$ 는 다음과 같다.

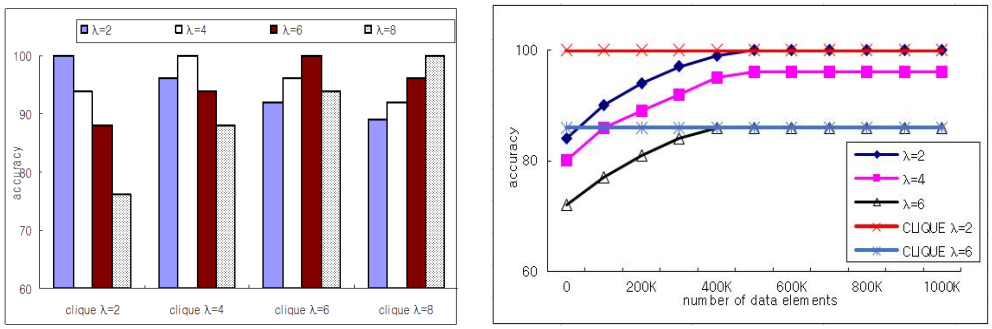
$$P_{min}(g)=\sum_{i=1}^{sl(v)} \log_h \frac{range(N_i)}{\lambda} \quad (5)$$

증명) 부분 공간 트리의 v번째 레벨에서 단위 격자셀 g를 찾으려면 최상위 노드에서 단위 격자셀 g까지의 경로에 있는 모든 격자셀도 단위 격자셀이어야 한다. 차원 데이터 공간 N_i의 격자셀 리스트에서 단위 격자셀을 생성하는 데 필요한 반복 분할 작업의 최소 횟수는 각 분할에서 격자셀을 h개의 격자셀로 나누기 때문에 분할 수는 $\log_h \frac{range(N_i)}{\lambda}$ 이다. 경로를 따라 생성되므로 각 격자셀 리스트에서 단위 격자셀을 만드는 데 필요한 분할 작업의 수를 누적해야 한다. v차원의 단위 격자셀에 대한 격자셀 리스트 $sl(v)$의 수는 v차원까지 조합에서 $sl(v)=\sum_{j=1}^{v} {}_vC_j$ 이다. 따라서 v 번째 수준에서 단위 격자 셀을 생성하는 최소 분할 수는 수식 (5)와 같이 $P_{min}(g_v)=\sum_{i=1}^{sl(v)} \log_h \frac{range(N_i)}{\lambda}$ 가 된다.

4. 실험

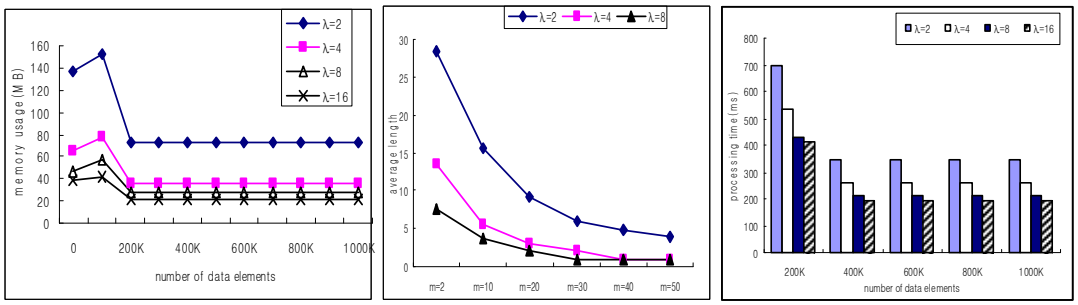
제안된 방법의 성능을 분석하기 위해 가우시안 분포의 데이터 생성기에 의해 40차원 데이터 세트를 생성한다 [9,10]. 데이터 집합에는 평균 차원이 15인 10개의 부분 공간 클러스터가 포함하며, 이들의 분포 평균 및 표준 편차는 무작위로 선택하였다. 두 개의 병합, 분할 지지 임계값 S_{mer} 및 S_{par}는 미리 정의된 최소 지지 S_{min}에 상대적으로 할당하였다. 대부분의 실험 조건은 다르게 지정하지 않는 한 S_{min}=0.01, S_{mer}=0.1xS_{min}, S_{par}=0.8xS_{min}, λ=2, m=40 및 h=4이다. 모든 실험에서 데이터 항목은 온라인 데이터 스트림의 환경을 시뮬레이션하기 위해 순서대로 하나씩 처리된다.

제안한 방법의 정확도는 그림 2와 같다. CLIQUE[14]는 잘 알려진 기존의 유한 데이터 세트에 대한 격자 기반 부분 공간 클러스터링 알고리즘으로 제안된 방법의 정확도를 측정하는 척도로 사용된다. 정확도는 제안하는 방법에 의해 찾은 부분 공간 클러스터의 데이터 항목 수 대비 CLIQUE에 의해 동일한 부분 공간 클러스터로 찾아진



(a) Accuracy comparison (b) Accuracy variation

Fig. 2. Accuracy camparison with CLIQUE



(a) Memory usage (b) Average length of grid-cell lists (c) Processing time

Fig. 3. Performance evaluation

데이터 항목 수로 측정한다. 그림 2-(a)는 4개의 서로 다른 λ 값에 대해 제안된 방법의 정확도를 보여준다. 제안하는 방법의 λ 값이 CLIQUE의 값과 같을 때 두 방법의 정확도는 동일하다. 그림 2-(b)는 새로운 데이터 항목이 생성될 때 정확도의 변화를 보여준다. 제안한 방법으로 구한 부분 공간 클러스터의 정확도는 λ=2일 때 CLIQUE의 클러스터링 결과와 비교하여 측정하였다. 부분 공간 클러스터링의 초기 단계에서 단위 격자 셀을 찾기 위해 많은 분할 작업이 발생하므로 제안 방법의 정확도가 상대적으로 낮다. 그러나 연속적인 분할 과정을 통해 단위 격자 셀을 찾을수록 정확도가 점차 높아진다.

그림 3-(a)는 메모리 사용량을 보여준다. 부분 공간 클러스터링의 초기 단계에서는 각 격자 셀의 지지도가 너무 민감하게 변하기 때문에 많은 분할 작업이 수행된다. 그러나 메모리 사용량은 초기 단계 이후에 안정화된다. 메모리 사용량은 λ 값에 비례하는 것을 확인할 수 있다. 그림 3-(b)는 부분 공간 트리에서 격자셀 리스트의 평균 길이를 보여준다. λ 값이 감소할수록 밀집 영역의 격자 셀의 수가 증가하여 격자셀 리스트의 평균 길이가 짧아진다. 그림 3-(c)는 제안된 방법의 처리 시간을 보여준다. 각 x축 단위 내의 데이터 항목당 평균 처리 시간을 표시하였다. 초기에 많은 파티셔닝 작업이 자주 발생하면 처리 시간이 늘어나지만 데이터 스트림을 진행하면서 안정화되어 다시 줄어드는 것을 볼 수 있다.

5. 결론

데이터 스트림의 차원 수가 증가함에 따라 부분 공간 클러스터링은 스트림의 부분 공간에서 관심 클러스터를 분석하는 데 유용하다. 그러나 기존의 부분 공간 클러스터링 방법은 가능한 모든 후보 부분 공간 클러스터를 생성하고 각 후보에 대해 전체 데이터 항목을 반복적으로 검사하는 방법으로 막대한 처리시간과 메모리 자원을 사용한다.

이러한 이유로 온라인 데이터 스트림 처리에는 적합하지 않다. 이 논문은 데이터 스트림에 대한 부분 공간 클러스터링 방법을 제시한다. 본 논문에서는 클러스터를 찾기 위해 셀 트리의 메모리 사용량을 제어하는 적응적 방법을 제안하였고, 이 방법을 확장하여 본 논문의 부분 공간 트리에서도 동적으로 필요한 부분 공간의 격자셀 리스트만 확장하여 클러스터링을 수행한다. 결과적으로 단위 격자 셀 λ의 정밀도를 가진 부분 공간 클러스터를 분포 정보 기반으로 동적 분할과정을 통해 찾아감으로 데이터 스트림에서 효율적으로 부분 공간 클러스터를 탐색할 수 있다. 실험에서 초기 분할과정에서 메모리 사용량이 높았지만, 진행되면서 최대 45% 가량의 자원으로 수행하는 것을 확인하였다. 향후에는 실제 처리속도 이상의 데이터 스트림에서 복잡한 데이터 마이닝을 수행하는 경우, 특히 클러스터 결과가 시간에 따라 계속 변화하는 경우 본 연구의 방법을 확장하여 실세계의 노이즈가 많은 데이터에서도 효율적인 방법을 연구할 것이다.

REFERENCES

[1] M. Garofalakis, J. Gehrke & R. Rastogi. (2002) Querying and mining data streams: you only get one look. *In the tutorial notes of the 28th Int'l Conference on Very Large Databases*, Hong Kong. DOI:10.1145/564691.564794

[2] Joong Hyuk Chang & Won Suk Lee. (2006). Finding frequent itemsets over online data streams. *Information & Software Technology*, 48(7), 606-618. DOI: 10.1016/j.infsof.2005.06.004

[3] Mohamed Medhat Gaber, Arkady B. Zaslavsky & Shonali Krishnaswamy. (2005). Mining data streams: a review. *SIGMOD Record 34(2)*, 18-26. DOI: 10.1145/1083784.1083789

[4] Ming Hua, Jian Pei & Xuemin Lin. (2011). Ranking queries on uncertain data. *The International Journal on Very Large Data Bases,* 20(1), 129-153. DOI: 10.1007/s00778-010-0196-4

[5] Jie Zhao, Xiaowen Li & Peiquan Jin. (2012). A Time-Enhanced Topic Clustering Approach for News Web Search. *Int. Journal of Database Theory and Application*, 5(4), 1-10.

[6] Chun-Hung Cheng, Ada Waichee Fu & Yi Zhang. (1999). Entropy-based subspace clustering for mining numerical data. *In Proceedings of the fifth ACM SIGKDD International Conference on Knowledge discovery and data mining*, 84-93. DOI: 10.1145/312129.312199

[7] Tang MingJing, Li Tong, Zhu Rui & Ma ZiFei. (2021). A Cluster Analysis Method of Software Development Activities Based on Event Log. *Recent Advances in Computer Science and Communications, 14(6).* 1843-1851. DOI: 10.2174/2666255813666191204144931

[8] Hans-Peter Kriegel, Peer Kroger, Matthias Renz & Sebastian Wurst. (2006). Generic Framework for Efficient Subspace Clustering of High-Dimensional Data. *In Proceedings of the Fifth IEEE International Conference on Data Mining*, 250-257. DOI: 10.1109/ICDM.2005.5

[9] Nam Hun Park & Won Suk Lee. (2007). Cell trees: An Adaptive Synopsis structure for clustering multi-dimensional on-line data streams. *J. Data & Knowledge Engineering, 63(2),* 528-549.
DOI: 10.1016/j.datak.2007.04.003

[10] O'callaghan, L., Mishra, N., Meyerson, A., Guha, S., & Motwani, R. (2002). Streaming-data algorithms for high-quality clustering. *In Proceedings 18th International Conference on Data Engineering,* 685-694.
DOI: 10.5555/876875.878995

[11] Charu C. Aggarwal, Jiawei Han, Jianyong Wang & Philip S. Yu. (2003). A Framework for Clustering Evolving Data Streams. *In Proc. VLDB 29th.*
DOI: 10.1016/B978-012722442-8/5 0016-1

[12] Mohamed Medhat Gaber. (2011). Advances in data stream mining. *Data Mining and Knowledge Discovery*, 2(1).
DOI: 10.1002/widm.52

[13] Eoin Martino Grua, Mark Hoogendoorn, Ivano Malavolta, Patricia Lago & A.E. Eiben. (2019). CluStreamGT Online Clustering for Personali -zation in the Health Domain. *IEEE/WIC/ ACM International Conference on Web Intelligence*.

[14] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos & Prabhakar Raghavan. (1998). Automatic subspace clustering of high dimensional data for data mining applications. *In Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, 94-105.
DOI: 10.1145/276305.276314

[15] Mohammed Oualid Attaoui, Hanene Azzag, Mustapha Lebbah & Nabil Keskes. (2020). Subspace data stream clustering with global and local weighting models, *Neural Computing and Applications, 33*, 3691-3712.
DOI: 10.1007/s00521-020-05184-z

박 남 훈(Nam Hun Park) [정회원]

· 2000년 2월 : 연세대학교 기계전자공학부 컴퓨터과학 전공(공학사)
· 2002년 2월 : 연세대학교 컴퓨터과학 (공학석사)
· 2007년 8월 : 연세대학교 컴퓨터과학 (공학박사)
· 2008년 9월 ~ 2010년 2월 : Worcester Polytec. Inst. Research Associate
· 2010년 3월 ~ 현재 : 안양대학교 융합소프트웨어학과 교수
· 관심분야 : 데이터 마이닝, 데이터 스트림
· E-Mail : nmhnpark@anyang.ac.kr