윈도우-조인: 이원성 기반 서브시퀀스 매칭을 위한 최적의 방법

김 상욱*, 박 대현*, 이 헌길*, 김 만순*, 박 정일*

강원대학교 컴퓨터정보통신공학부*

Window-Join: An Optimal Way to Process Duality-Based Subsequence Matchi

Sang-Wook Kim*, Dae-Hyun Park*, Heon-Gil Lee*, Man-Soon Kim*, Jung-Il Park*

Division of Computer, Information, and Communications Engineering
Kangwon National University*

요약문

본 논문에서는 시계열 데이터베이스에서 서브시퀀스 매칭을 효과적으로 처리하는 방안에 관하여 논의한다. 본 논문에서는 먼저, 기존의 이원성 기반 서브시퀀스 매칭 기법에서 발생하는 성능상의 문제점들을 지적하고, 이들을 해결할 수 있는 방법을 제시한다. 제안된 기법은 서브시퀀스 매칭 시 요구되는 인덱스 검색을 윈도우-조인이라는 일종의 공간 조인 문제로 새롭게 해석하는 것에서 출발한다. 제안된 기법에서는 효과적인 윈도우-조인의 처리를 위하여 질의 윈도우 점들을 위한 R*-트리를 주기억장치 내에 on-the-fly로 구성하는 방법을 사용한다. 또한, 데이터 윈도우 점들을 위한 디스크 상의 R*-트리와 질의 윈도우 점들을 위한 주기억장치 상의 R*-트리를 효과적으로 조인할 수 있는 새로운 알고리즘을 제안한다. 제안된 기법은 R*-트리 페이지들을 착오 채택 없이 단 한번만 디스크로부터 액세스하므로 디스크 액세스 측면에서 이원성 기반 서브시퀀스 매칭을 위한 최적의 기법이다.

1. 서론[1)]

시계열 데이터베이스(time-series databases)란 각 객체의 변화되는 값들의 연속으로 구성된 데이터 시퀀스(data sequences)들의 집합이며, 유사 검색(similarity search)이란 주어진 질의 시퀀스(query sequence)와 변화의 추세가 유사한 시퀀스들을 검색하는 연산이다[Agr93][Fal94][Raf97]. 유사 검색은 시계열 데이터베이스를 기반으로 하는 데이터 마이닝(data mining) 및 데이터 웨어하우징(data warehousing) 분야에서 중요한 연산으로 사용된다[Che96][Raf97][Loh00][Kim01].

유사 검색은 크게 전체 매칭(whole matching)과 서브시퀀스 매칭(subsequence matching)으로 분류된다[Agr93][Fal94][Par01]. 전체 매칭은 모든 데이터 시퀀스들과 질의 시퀀스의 길이가 항상 동일하다는 전제하에 질의 시퀀스와 유사한 시퀀스를 데이터베이스로부터 검색한다. 반면, 부분 매칭은 다양한 길이의 데이터 시퀀스들이 존재하는 것을 허용하며, 데이터베이스로부터 질의 시퀀스와 유사한 서브시퀀스를 포함하는 시퀀스와 그 시퀀스 내에서의 유사 서브시퀀스의 오프셋을 검색한다. 이와 같이, 서브시퀀스 매칭은 길이에 대한 제약이 없으므로 다양한 실제 응용 분야에서 널리 사용된다.

본 논문에서는 이러한 서브시퀀스 매칭을 효과적으로 처리하는 방안에 관하여 논의하고자 한다.

2. 관련 연구

본 장에서는 본 논문에서 해결하고자 하는 문제를 공식적으로 정의하고, 이에 대한 해결 방안을 제시한 기존의 연구 결과를 요약한다.

2.1. 문제 정의

시계열 데이터베이스 D, 길이 n의 질의 시퀀스 Q = (q[0], q[1], ..., q[n-1]), 그리고, 유사 허용치 ε이 주어질 때, 서브시퀀스 매칭 문제는 다음과 같이 정의된다. D에 저장된 길이 N의 임의의 시퀀스 S = (s[0], s[1], ..., s[N-1]) 내에 존재하는 길이 n의 임의의 서브시퀀스 X = (x[0] x[1], ..., x[n-1])가 다음 조건을 만족하면, <S, S내 X의 오프셋>을 반환한다.

$$d(X,Q) \le \varepsilon, \text{ 여기서 } d(X,Q) = \sqrt{\sum_{i=0}^{n-1} (x[i]-q[i])^2} \quad 2)$$

2.2. FRM [Fal94]

참고 문헌 [Fal94]에서는 서브시퀀스 매칭을 위한 좋은 해결 방안을 제안하였다. 본 논문에서는 참고 문헌 [Moo01]의 명칭을 따라 이 기법을 FRM이라 부른다. FRM에서는 미리 고정된 크기 w를 갖는 윈도우(window) 개념을 이용한다.

먼저, 각 시퀀스로부터 모든 가능한 위치에서 시작되는 길이 w의 슬라이딩 윈도우(sliding window)들을 추출하고, 각 윈도우를 이산 퓨리에 변환(discrete Fourier transform: DFT)을 이용하여 저차원 공간상의 점으로 변환한다. 본 논문에서는 이 점을 윈도우 점(window point)라 정의한다. 인덱싱의 대상이 되는 윈도우 점들의 수가 매우 많으므로, FRM에서는 이들을 다수의 점들을 포함하는 최소 포함 사각형(minimum bounding rectangle: MBR)들로 구성한 후, 이 MBR들을 다차원 인덱스(multidimensional index)의 하나인 *R*-트리[Bec90]*에 저장한다[3)].

서브시퀀스 매칭을 위해서는 질의 시퀀스로부터 크기 w의 디스조인트 윈도우(disjoint window)들을 추출하고, 윈도우들을 DFT하여 저차원 공간상의 윈도우 점들로 변환한다. 각 윈도우 점에 대하여 $\varepsilon/p^{1/2}$를 허용치로 갖는 범위 질의를 R*-트리 상에서 수행한다. 이러한 범위 질의의 결과로 얻어진 데이터 윈도우 점들을 조사함으로써 최종 결과에 포함될 가능성이 높은 후보 서브시퀀스(candidate subsequence)들을 파악한다. 그 다음, 이 후보 서브시퀀스들을 포함하는 각 시퀀스를 디스크로부터 액세스하여 후보 서브시퀀스와 질의 시퀀스와의 유클리드 거리를 실제로 계산한다. 끝으로, 이러한 착오 채택(false alarm)[Agr94]을 제거시킨 나머지 서브시퀀스들을 최종 결과로 반환한다.

2.3. Dual-Match [Moo01]

FRM에서는 인덱싱을 위한 저장 공간의 오버헤드를 줄이기 위하여 개별적 윈도우 점들 대신 다수의 윈도우 점들을 포함하는 MBR들을 R*-트리 내에 저장한다. 이러한 MBR 내부에는 죽은 공간(dead space)[Bec90]이 존재하게 되므로, 이로 인하여 후보 서브시퀀스의 착오 채택이 발생되며, 이것은 처리 성능 저하로 직결된다[Moo01]. 참고 문헌 [Moo01]에서는 이러한 문제점을 해결하기 위한 방법으로서 이원성 기반 서브시퀀스 매칭(duality-based subsequence matching: Dual-Match)을 제안하였다.

Dual-Match에서는 데이터 시퀀스로부터 슬라이딩 윈도우를 추출하고

---

1) 본 논문은 한국학술진흥재단 선도연구자 연구비 지원에 의하여 연구되었습니다. (과제번호: KRF-2000-041-E00258)

2) 즉, 유클리드 거리(Euclidean distance)가 주어진 허용치 ε 이하인 두 서브시퀀스 X와 Q를 유사하다고 간주한다.

3) 현재, 시계열 데이터베이스에서 가장 널리 사용되는 다차원 인덱스는 R*-트리이다. 따라서 본 논문에서는 다차원 인덱스와 R*-트리라는 용어를 혼용한다.

질의 시퀀스로부터 디스조인트 윈도우를 추출하는 FRM과는 반대로 데이터 시퀀스로부터는 디스조인트 윈도우를 추출하고 질의 시퀀스로부터는 슬라이딩 윈도우를 추출하는 방식을 사용한다. 이와 같은 역할 교환을 통하여 Dual-Match에서는 R*-트리에 저장할 윈도우들의 수를 FRM의 약 1/w로 줄일 수 있다. 이 결과, MBR들을 저장하는 FRM과는 달리 Dual-Match에서는 윈도우 점 자체를 R*-트리에 저장하는 것이 가능해진다. 따라서 MBR 내의 죽은 공간으로 인한 후보 서브시퀀스의 착오 채택이 발생되지 않으므로, 처리 성능이 크게 개선된다. 참고 문헌 [Moo01]의 성능 평가 결과에 의하면, Dual-Match의 검색 성능은 FRM과 비교하여 최대 430배까지 개선되는 것으로 나타났다.

3. 연구 동기

Dual-Match에서는 질의 시퀀스로부터 슬라이딩 윈도우를 추출하므로, FRM에서와는 달리 매우 많은 수의 질의 윈도우 점들이 생성된다. 예를 들어, 참고 문헌 [Moo01]의 실험에서 사용한 것과 같이 질의 시퀀스 길이가 1024이고 윈도우 길이가 256인 경우에 생성되는 윈도우 점들의 수는 769(= 1024-256+1)개이다. 본 연구는 이와 같이 많은 질의 윈도우 점들이 생성됨으로 인한 Dual-Match의 성능상의 문제점을 파악하는 것으로부터 출발하였다[4].

참고 문헌 [Moo01]에서는 이러한 질의 윈도우 점들을 이용하여 R*-트리를 검색하는 세 가지 방법을 제시하고 있다[5]. 본 장에서는 연구 동기로서 각 방법에서 나타나는 성능상의 문제점들을 각각 지적한다.

방법 (1): 모든 윈도우 점들에 대하여 각각 영역 질의를 수행

이 방법에서는 윈도우 점들의 개수만큼의 인덱스 검색이 발생하게 된다. 위 예의 경우, R*-트리가 총 769회 검색되어야 함을 의미한다. 특히 공간상에서 인접한 많은 윈도우 점들에 대한 인덱스 검색은 동일한 인덱스 페이지를 여러 번 디스크로부터 액세스하도록 하는 결과를 초래하게 된다. 따라서 이러한 다차원 인덱스 검색 비용은 매우 심각하다.

방법 (2): 모든 윈도우 점들을 포함하는 단 하나의 질의-MBR에 대하여 영역 질의를 수행

이 방법에서는 단 한번의 인덱스 검색만이 발생하게 된다. 따라서 방법 (1)에서와 같이 같은 인덱스 페이지를 다수 액세스하는 문제는 발생하지 않는다. 반면, 이 방식에서는 영역 질의에서 사용되는 MBR(간략히, **질의-MBR**) 내에 존재하는 죽은 영역으로 인한 후보 서브시퀀스의 착오 채택이 발생할 수 있다. Dual-Match에서는 모든 질의 윈도우 점들이 주기억장치 내에서 관리된다는 점을 활용하여 이 문제를 완화시킨다[Moo01]. 즉, 인덱스 검색을 통하여 후보로 반환되는 각 데이터 윈도우 점이 실제로 질의 윈도우 점과 $\varepsilon/p^{1/2}$이하의 거리 이내에 존재하는가를 인덱스 검색 단계에서 직접 확인하는 것이다. 참고 문헌 [Moo01]에서는 이러한 과정을 인덱스 수준 여과(index-level filtering)라 정의한다. 이 결과, 방법 (1)과 방법 (2)에서 인덱스 검색 결과로 반환되는 후보 서브시퀀스들의 수는 차이가 없게 된다.

그러나 이 방법은 다음과 같은 두 가지 문제점을 가진다.

첫째, 방법 (1)에서는 액세스되지 않던 인덱스 페이지들을 추가로 액세스하는 경우가 발생한다. 이것은 질의-MBR 내의 죽은 영역으로 인하여 그 어떤 질의 윈도우 점과도 $\varepsilon/p^{1/2}$ 초과되는 거리에 존재하는 R*-트리 내의 MBR이 $\varepsilon/p^{1/2}$이내의 거리로 간주됨으로써 발생하는 현상이다. 본 논문에서는 이러한 문제를 **인덱스 단계의 착오 채택(index-level fals alarm)**이라 정의한다. 모든 질의 윈도우 점들을 포함하는 하나의 질의-MBR은 매우 크므로 그 내부의 죽은 영역 또한 매우 크다. 따라서 인덱스 단계의 착오 채택으로 인한 추가적인 인덱스 페이지 액세스의 오버헤드는 심각하다.

둘째, 인덱스 수준 여과에서 발생하는 CPU 오버헤드이다. 질의-MBR에 의하여 반환되는 **각각의** 후보 데이터 윈도우 점은 최악의 경우 **모든** 질의 윈도우 점들과의 비교를 수행하는 인덱스 수준 여과를 거쳐야 한다. 앞의 예에서의 경우, 각 후보 윈도우 점은 평균 385(769/2)개의 질의 윈도우 점들과의 비교 연산을 수행해야 함을 의미한다. 큰 질의-MBR에 의하여 반환되는 후보 윈도우 점들의 수는 매우 많으므로, 이러한 CPU 오버헤드 역시 전체 처리 성능을 떨어뜨리는 중요한 원인이 된다.

방법 (3): 다수의 윈도우 점들을 포함하는 다수의 질의-MBR들에 대하여 각각 영역 질의를 수행

이 방법은 방법 (1)과 방법 (2)의 절충안으로서, 형성되는 질의-MBR들의 개수만큼의 인덱스 검색이 발생하게 된다. 질의-MBR을 생성하는 방법으로는 모든 질의-MBR이 같은 수의 윈도우 점들을 포함하도록 하는 방법과 질의-MBR의 수와 죽은 영역의 크기를 동시에 최소화하는 휴리스틱 방법 등 참고 문헌 [Fal94]에서 제시한 방식을 고려할 수 있다.

이러한 방법은 방법 (1)과 방법 (2)에서 발생하는 각 문제점을 완화시키는 경향이 있으나, 두 방법에서 발생하는 문제들(같은 인덱스 페이지를 다수 액세스하는 문제, 인덱스 단계의 착오 채택 문제, 인덱스 수준 여과에서 발생하는 CPU 오버헤드 문제)을 모두 가진다. 따라서 근원적인 해결책이 될 수 없다.

4. 제안하는 기법

본 장에서는 제 3장에서 지적한 문제점들을 해결할 수 있는 새로운 서브시퀀스 매칭 기법을 제안한다. 제안하는 기법은 Dual-Match를 기반으로 한다. 따라서 데이터 시퀀스들로부터 슬라이딩 윈도우들을 추출함으로써 R*-트리 인덱스를 구성하고, 서브시퀀스 매칭의 처리 시에는 질의 시퀀스로부터 디스조인트 윈도우들을 추출한다. 제안하는 기법에서 추구하는 궁극적인 목표는 서브시퀀스 매칭 시 발생하는 디스크 액세스 횟수를 최소화하는 것이다.

4.1. 문제의 재해석

본 논문에서는 서브시퀀스 매칭에서 후보 윈도우를 찾는 과정을 아래와 같이 윈도우-조인(window-join) 문제로 재해석한다.

정의 1: 윈도우-조인(window-join)

데이터 윈도우 점들의 집합 D_w와 질의 윈도우 점들의 집합 Q_w를 대상으로 상호 $\varepsilon/p^{1/2}$이하의 거리를 가지는 <데이터 윈도우 점, 질의 윈도우 점>의 쌍들의 집합을 찾는 공간 조인 문제.

□

서브시퀀스 매칭에 관한 기존의 다양한 연구가 있었으나, 저자들이 아는 한 이와 같이 후보 윈도우들을 찾는 문제를 이와 같이 공간 조인(spatial join)의 관점에서 해석한 시도는 없었다. 이러한 해석이 가지는 중요한 의미는 이 문제에 대한 새로운 관점에서의 해결책을 마련할 수 있다는 데 있다. 본 연구에서는 이러한 점에 착안하여 D_w와 Q_w간의 공간 조인을 효과적으로 수행하는 새로운 기법을 고안하고자 한다.

4.2. 윈도우-조인의 처리 방안

두 개의 데이터 집합에 대한 공간 조인을 효과적으로 처리하는 문제는 GIS 및 공간 데이터베이스 분야의 중요한 문제의 하나로 간주되어 왔다[Bri93][Hua97]. 현재까지 알려진 가장 효과적인 방법의 하나는 대상이 되는 두 집합에 다차원 인덱스가 존재한다는 가정 하에 인덱스 기반 공간 조인을 수행하도록 하는 것이다[Bri93][Hua97][Son99].

그러나 현재 우리가 직면한 상황에서는 D_w에는 R*-트리가 존재하지만 Q_w에는 다차원 인덱스가 존재하지 않는다. 따라서 참고 문헌 [Bri93][Hua97]에서와 같은 효과적인 공간 조인을 수행하는데, 어려움이 있다. 본 연구에서는 이를 극복하기 위한 방안으로서 Q_w를 위한 다차원 인덱스를 주기억장치 내에 on-the-fly로 생성하는 방법을 사용한다. 물론 on-the-fly 인덱스의 생성은 질의 처리를 위한 추가의 비용을 요구한다. 그러나 Q_w가 D_w에 비하여 훨씬 작으며, 또한 Q_w내의 윈도우 점들이 이미 주기억장치 내에서 관리되므로 on-the-fly 인덱스의 생성 비용은 상대적으로 미미하다. 또한, 이로 인하여 전체 윈도우-조인 처리 시 발생하는 디스크 액세스를 줄일 수 있으므로, 이러한 on-the-fly 인덱스의 생성은 정당화 될 수 있다. 본 연구에서는 on-the-fly 인덱스로서 주기억장치 R*-트리를 사용한다. 인덱스의 생성은 Q_w내의 윈도우 점을 반복적으로 주기억장치 내의 R*-트리 내에 삽입함으로써 가능하다. 인덱스 생성 효율의 극대화하기 위하여 벌크 로드(bulk load)[Kam93] 기법을 활용할 수도 있다.

이제 D_w와 Q_w에 모두 R*-트리가 존재하므로, 참고 문헌 [Bri93][Hua97 등에 제안된 기존의 기법을 활용함으로써 공간 조인을 효과적으로 처리할 수 있다. 기존의 공간 조인 기법들은 두 R*-트리가 모두 디스크 상에 존재한다는 것을 가정한다. 따라서 양쪽의 R*-트리를 동등하게 대우함으로써 양쪽에서 발생하는 디스크 액세스 수를 동시에 최적화 하려고 시도한다. 그러나 전체 공간 내의 점들은 다양한 형태로 분포하므로, 이러한 시도의 결과 동일한 인덱스 페이지가 두 번 이상 액세스되는 현상이 발생된다[Bri93][Hua97]. 반면, 현재 우리의 윈도우-조인에서는 D_w를 위한 R*-트리는 디스크 내에, 그리고 Q_w를 위한 R*-트리는 주기억장치 내에 존재한다. 따라서 본 연구에서는 윈도우-조인의 고유한 특징을 활용함으로써 최적의 성능을 제공하는 새로운 공간 조인 기법을 제시할 수 있다.

제안하는 공간 조인 기법에서는 D_w를 위한 디스크 상의 R*-트리 인덱

4) 본 장에서 지적하는 대부분의 현상이 FRM에서도 동일하게 발생한다. 그러나 FRM과 비교하여 Dual-Match에서 생성되는 질의 윈도우 점들의 수가 훨씬 많으므로 큰 성능상의 문제로 부각되는 것이다.

5) 이들 중, 실제 실험에서 채택한 것은 방법 (2)이다[Moo01].

스 I_D를 외부 릴레이션(outer relation), Q_w를 위한 주기억장치 상의 인덱스 I_Q를 내부 릴레이션(inner relation)으로 간주한다. 제안하는 공간 조인 기법의 수행 과정을 간략히 요약하면 다음과 같다. (1) I_D의 루트 페이지 내에 있는 각 MBR에 대하여 이를 $\varepsilon/p^{1/2}$만큼 확장한 MBR을 사용하여 I_Q에 영역 질의를 수행한다. 이 MBR을 e-MBR (enlarged MBR)이라 부른다. (2) I_Q에 대한 영역 질의의 결과, I_Q내의 그 어떤 질의 윈도우 점과도 $\varepsilon/p^{1/2}$ 이상 떨어진 것이 확인된 I_D의 MBR에 대해서는 그 하위 서브트리를 이후의 고려 대상에서 완전히 제외시킨다. (3) 반면, I_Q에 대한 영역 질의의 결과, I_Q내의 어떤 질의 윈도우 점과 $\varepsilon/p^{1/2}$ 이내에 있다고 판단된 MBR(c-MBR: confirmed MBR)에 대해서는 I_D내의 그 하위 단계 페이지에 대하여 이러한 작업을 재귀적으로 반복한다.

윈도우 조인은 I_D와 I_Q 모두에 대하여 깊이 우선 탐색(depth first search) 방식으로 진행된다. 또한, 각 e-MBR을 이용하여 I_Q에 대한 영역 질의를 수행할 때, CPU-최적화를 위하여 다음과 같은 방식을 사용한다. 영역 질의 수행 중, e-MBR이 I_Q의 어떤 MBR을 완전히 포함하는 경우, 즉시 영역 질의를 중단하고 이 e-MBR을 c-MBR로 간주한다. I_D의 리프 페이지를 액세스한 경우에는 이와 대응되는 e-MBR에 대한 영역 질의 결과 반환된 I_Q 의 윈도우 점들과 이 페이지 내의 윈도우 점들을 조인함으로써 최종 후보 윈도우 쌍들의 집합을 구하게 된다. 제안된 공간 조인에 대한 보다 상세한 알고리즘은 참고 문헌 [Kim00]에 제시되어 있다.

4.3. 제안된 윈도우-조인 기법의 장단점

본 절에서는 제안된 윈도우-조인 처리 성능에 관하여 논의한다.

참고 문헌 [Bri93][Hua97]의 기법들에서와는 달리, 제안된 기법에서는 I_D를 외부 릴레이션으로 간주하므로 윈도우-조인의 처리 시 액세스되는 I_D 내의 각 인덱스 페이지는 디스크로부터 단 한번만 액세스된다. 따라서 제 3장에서 언급한 Dual-Match의 인덱스 검색 방법 (1)과 (3)에서와 같은 동일한 인덱스 페이지를 두 번 이상 액세스하는 현상은 발생하지 않는다.

또한, 윈도우-조인의 처리 시, 디스크로부터 액세스되는 I_D 내의 인덱스 페이지는 I_Q에 대한 영역 질의를 통하여 액세스 필요성이 확인된 것이므로 인덱스 단계의 착오 채택이 전혀 발생하지 않는다. 따라서 제 3장에서 언급한 Dual-Match의 인덱스 검색 방법 (2)와 (3)에서와 같은 인덱스 단계 착오 채택으로 인한 디스크 액세스의 오버헤드가 제거된다.

끝으로, 인덱스 여과 과정에서 발생하는 윈도우 점들간의 비교 횟수가 크게 줄어든다. Dual-Match의 인덱스 검색 방법 (2)와 (3)에서는 각 후보 윈도우 쌍과 모든 질의 윈도우 점들을 비교해야 한다. 반면, 제안하는 윈도우-조인 기법에서 각 후보 윈도우 쌍은 이를 포함하는 상위 단계 e-MBR을 이용한 영역 질의 결과로 반환되는 질의 윈도우 점들만을 비교 대상으로 한다. 따라서 CPU 성능이 크게 개선된다.

윈도우-조인의 처리 과정에서 I_D 내의 얼마나 많은 인덱스 페이지들이 상위 단계에서 cut-off 되는가를 파악하기 위하여 다음과 같은 선행 실험을 수행하였다. 이 실험에서는 평균 길이가 231인 545개의 시퀀스들로 구성되는 미국의 S&P 500 주식 데이터를 사용하였다. 또한, 윈도우의 길이는 64를 사용하였으며, 질의 시퀀스의 길이는 256을 사용하였다. 데이터 시퀀스와 질의 시퀀스로부터 각각 디스조인트 윈도우와 슬라이딩 윈도우를 추출하고, 각각을 DFT 변환한 후, 앞쪽의 두 DFT 계수로부터 실수부와 허수부 총 4개의 특징 중 세 개만을 골라 인덱싱을 위한 특징으로 선택하였다. 그림 1은 데이터 윈도우 점들이 분포하는 전체 특징 공간상에서 질의 시퀀스로부터 추출된 질의 윈도우 점들이 분포되는 상황을 도면화 한 것이다. 질의 윈도우 점들은 전체 특징 공간상에서 **극히 일부**에 분포됨을 볼 수 있다. 이는 윈도우-조인 처리 시, I_D의 루트 단계에서부터 수많은 MBR들이 이후의 처리 대상에서 제외됨을 나타내는 것이다.

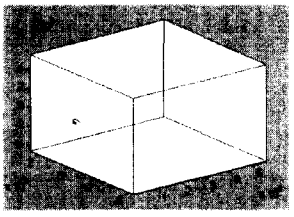

그림 1. 전체 시퀀스 윈도우들을 위한 특징 공간상에서의 질의 윈도우 점들의 분포.

제안된 기법에서 요구하는 추가 작업은 (1) R*-트리의 on-the-fly 구성 및 (2) I_D 내의 각 e-MBR을 대상으로 하는 I_Q에 대한 영역 질의 처리이다 그러나 이 두 작업은 모두 주기억장치 내에서 처리되며, 제 4.2절에서 설명한 CPU-최적화를 수행하므로 그 성능 저하 효과는 앞서 언급한 성능 개선 효과들과 비교하여 미미하다.

5. 결론

본 논문에서는 윈도우-기반 서브시퀀스 매칭을 최적으로 처리할 수 있는 새로운 기법을 제안하였다. 본 논문에서는 서브시퀀스 매칭 시 요구되는 인덱스 검색을 윈도우-조인 문제로 해석함으로써 이 문제에 대한 새로운 해결책을 제시하였다. 제안된 기법은 데이터 윈도우를 위한 R*-트리 내에 페이지들을 착오 채택 없이 단 한번만 디스크로부터 액세스하도록 한다. 따라서 디스크 액세스 측면에서 최적의 기법이라 할 수 있다. 현재, 제안된 기법이 가지는 성능 개선 효과를 정량적으로 분석하기 위하여 다양한 실험을 수행하고 있다.

6. 참고 문헌

[Agr93] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient Similarity Search in Sequence Databases," In Proc. Int'l. Conf. on Foundations of Data Organization and Algorithms, FODO, pp. 69-84, Oct. 1993.

[Bec90] N. Beckmann et al., "The R*-tree: An Efficient and Robu Access Method for Points and Rectangles," In Proc. Int'l. Conf on Management of Data, ACM SIGMOD, pp. 322-331, May 1990.

[Bri93] T. Brinkhoff, H.-P. Kriegel, and B. Seeger, "Efficient Processin of Spatial Joins Using R-Trees," In Proc. ACM Int'l. Conf. o Management of Data, ACM SIGMOD, pp. 237-246, 1993.

[Che96] M. S. Chen, J. Han, and P. S. Yu, "Data Mining: An Overview from Database Perspective," IEEE Trans. on Knowledge and Data Engineering, Vol. 8, No. 6, pp. 866-883, 1996.

[Fal94] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast Subsequence Matching in Time-series Databases," In Proc. Int Conf. on Management of Data, ACM SIGMOD, pp. 419-429, May 1994.

[Hua97] Y.-W. Huang, N. Jing, and E. A. Rundensteiner, "Spatial Joins Using R-trees: Breadth-First Traversal with Glob Optimizations," In Proc. Int'l. Conf. on Very Large Data Bases VLDB pp. 396-405, 1997.

[Kam93] I. Kamel and C. Faloutsos, "On Packing R-trees," In Proc. Int' Conf. on Information and Knowledge Management, ACM CIKM, pp. 490-499, 1993.

[Kim00] S. W. Kim, Window-Join: A New Paradigm that Optimizes Index Search for Subsequence Matching in Time-Serie Databases, 2000. (unpublished manuscript)

[Kim01] S. W. Kim, S. H. Park, and W. W. Chu, "An Index-Base Approach for Similarity Search Supporting Time Warping in Large Sequence Databases," In Proc. IEEE Int'l. Conf. on Dat Engineering, IEEE ICDE, pp. 607-614, 2001.

[Loh00] W. K. Loh, S. W. Kim, and K. Y. Whang, "Index Interpolation An Approach for Subsequence Matching Supporting Normalization Transform in Time-Series Databases," In Pro ACM Int'l. Conf. on Information and Knowledge Management, ACM CIKM, pp. 480-487, 2000.

[Moo01] Y. S. Moon, K. Y. Whang, and W. K. Loh, "Duality-Base Subsequence Matching in Time-Series Databases," In Proc. IEE Int'l Conf. on Data Engineering, IEEE ICDE, pp. 263-272, 2001.

[Par01] S. H. Park, S. W. Kim, and W. W. Chu, "Segment-Base Approach for Subsequence Searches in Sequence Databases", In Proc. ACM Int'l. Symp. on Applied Computing, ACM SAC, pp. 248-252, 2001.

[Raf97] D. Rafiei and A. Mendelzon, "Similarity-Based Queries for Time-Series Data," In Proc. Int'l. Conf. on Management o Data, ACM SIGMOD, pp. 13-24, 1997.

[Son99] J. W. Song, K. Y. Whang, Y. K. Lee, and S. W. Kim, "Spati Join Processing Using Corner Transformation," IEEE Trans. o Knowledge and Data Engineering, Vol. 11, No. 4, pp. 688-695, 1999.