시계열 데이터베이스에서 타임 워핑 하의 서브시퀀스 매칭의 성능 최적화[1)]

김만순*, 김상욱**
*강원대학교 정보통신공학과
**한양대학교 정보통신학부
e-mail: mansoon@kangwon.ac.kr, wook@hanyang.ac.kr

Optimization of Subsequence Matching Under Time-Warping in Time-Series Databases

Man-Soon Kim*, Sang-Wook Kim**
*Division of Computer, Information and Communications
Kangwon National University
**College of Information, and Communications
Hanyang University

요 약

본 논문에서는 시계열 데이터베이스에서 타임 워핑 하의 서브시퀀스 매칭을 효과적으로 처리하는 방안에 관하여 논의한다. 타임 워핑은 데이터베이스내 시퀀스들의 길이가 서로 다른 경우에도 유사한 패턴을 갖는 시퀀스들을 찾을 수 있도록 해 준다. 본 논문에서는 타임 워핑 하의 서브시퀀스 매칭을 위한 기존의 기본 처리 방식인 Naive-Scan의 CPU 처리 과정을 최적화하는 새로운 기법을 제안한다. 제안된 기법은 질의 시퀀스와 서브시퀀스들 간의 타임 워핑 거리들을 계산하는 과정에서 발생하는 중복 작업들을 사전에 제거함으로써 CPU 처리 성능을 극대화한다. 제안된 기법이 착오 기각을 발생시키지 않음과 Naive-Scan을 처리하기 위한 최적의 기법임을 이론적으로 규명한다. 또한, 다양한 실험을 통한 성능 평가에 의하여 제안된 최적화 기법이 가져오는 성능 개선 효과를 정량적으로 검증한다. 아울러, 제안된 기법이 기존의 여과 단계를 포함하는 방식인 LB-Scan과 ST-Filter의 후처리 단계에도 성공적으로 적용될 수 있음을 보인다.

1. 서론

시계열 데이터베이스(time-series database)란 객체의 변화되는 값들의 연속으로 구성된 데이터 시퀀스(data sequence)들의 집합이다[Agr93]. 시퀀스 매칭(sequence matching)이란 주어진 질의 시퀀스(query sequence)와 변화의 패턴이 유사한 시퀀스들을 시계열 데이터베이스로부터 찾아내는 데이터 마이닝(data mining) 및 데이터 웨어하우징(data warehousing) 분야의 중요한 연산이다[Agr93][Fal94].

시퀀스 매칭에 관한 기존의 연구에서는 길이 n의 시퀀스를 n 차원 공간상의 한 점으로 간주한다. 또한, 길이 n인 동일한 서로 다른 두 시퀀스 X(=[x_1, x_2, ... , x_n])와 Y(=[y_1, y_2, ... , y_n])간의 유사한 정도를 측정하는 척도로서 아래의 식과 같이 정의되는 거리 함수 $L_p(X,Y)$를 널리 사용한다. L_1은 맨하탄 거리(Manhattan distance), L_2는 유클리드 거리(Euclidean distance), L_∞은 대응되는 각 요소 값 쌍의 거리 중 최대 거리를 의미한다. 응용에서 주어진 허용치 ε보다 작은 $L_p(X,Y)$를 갖는 임의의 두 시퀀스 X, Y를 유사하다고 간주한다[Agr93][Fal94].

$$L_p(X,Y) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p}$$

L_p 거리 함수만을 이용한 시퀀스 매칭을 통해서는 사용자가 원하는 시퀀스들을 검색하지 못하는 경우가 빈번하게 발생한다. 따라서 응용 분야에 적합한 유사 모델(similarity model)을 적절하게 정의할 수 있도록 변환(transform)을 지원하기도 한다.

타임 워핑은 시퀀스내의 각 요소 값을 임의의 수만큼 반복시키는 것을 허용하는 변환이다[Yi98]. 타임 워핑 후의 두 시퀀스들 간의 거리를 타임 워핑 거리(time warping distance)라 한다. 두 시퀀스 S와 Q간의 타임 워핑 거리(time warping distance) D_{tw}는 다음과 같이 재귀적으로 정의된다[Yi98][Par00][Kim01]:

정의 1:

(1) $D_{tw}(\langle\rangle, \langle\rangle) = 0$,
(2) $D_{tw}(S, \langle\rangle) = D_{tw}(\langle\rangle, Q) = \infty$,
(3) $D_{tw}(S, Q) = (|L_p(First(S), First(Q))|^p + |\min(D_{tw}(S, Rest(Q)), D_{tw}(Rest(S), Q), D_{tw}(Rest(S), Rest(Q)))|^p)^{1/p}$

여기서, First(S)는 각각 S의 첫 번째 요소 s_1을 의미하며, Rest(S)는 s_1을 제외한 S의 나머지 요소들로 구성되는 시퀀스를 의미한다. $\langle\rangle$은 요소가 존재하지 않는 널 시퀀스(null sequence)를 의미한다. min은 세 개의 인자들 중 가장 작은 값을 가지는 것을 취하는 함수이다. L_p는 응용에서 적합한 것을 선택하여 사용할 수 있다. 본 논문에서는 현재 가장 널리 사용되는 맨해튼 거리(Manhattan distance) L_1을 기반으로 하는 타임 워핑 거리에 연구의 초점을 맞추고자 한다. □

타임 워핑 거리는 데이터베이스내의 시퀀스들의 길이가 서로 달라서 L_p 거리 함수를 이용하여 유사 정도를 직접 측정할 수 없는 경우에 매우 유용하다[Kim01]. 현재, 타임 워핑은 음성 인식 분야에서 널리 사용되고 있으며[Rab93], 심전도 데이터, 주가 데이터, 기온 데이터, 기업 성장률 데이터 등에도 유사한 방식으로 적용할 수 있다.

최근, 타임 워핑 하의 시퀀스 매칭에 관한 다양한 연구가 수행되어 왔다[Ber96][Yi98][Par00][Kim01][Par01]. 타임 워핑 하의 시퀀스 매칭의 처리를 위한 기존의 기법들은 크게 여과 단계(filtering step)와 후처리 단계(post processing step)로 구성된다. 여과 단계는 주어진 질의 시퀀스와 전혀 유사하지 않는 시퀀스들을 미리 제거함으로써 최종 결과에 포함될 가능성이 매우 높은 시퀀스들로만 구성되는 후보 집합(candidate set)을 구성하는 단계이다. 질의 시퀀스와 실제로

1) 본 연구는 2003년도 한양대학교 교내 연구비의 지원과 정보통신부 대학 IT 연구센터(센터명: 미디어서비스기술연구센터) 육성·지원사업의 연구 결과로 수행되었습니다.

유사한 시퀀스를 필터링 단계에서 후보 집합 내에 포함시키지 못하는 현상을 착오 기각(false dismissal)이라 한다. 반면, 질의 시퀀스와 유사하지 않은 일부의 시퀀스를 필터링 단계에서 후보 집합 내에 포함시키는 현상을 착오 채택(false alarm)이라 한다. 후처리 단계는 후보 집합에 속하는 각 시퀀스를 디스크로부터 액세스하여 이것이 질의 시퀀스와 실제로 유사한가의 여부를 판단함으로써 착오 채택을 제거하는 단계이다.

참고 문헌 [Ber96]에서는 여과 단계 없이 모든 시퀀스들 각각에 대하여 질의 시퀀스와의 타임 워핑 거리를 동적 프로그래밍(dynamic programming)을 이용하여 계산함으로써 타임 워핑 하의 시퀀스 매칭을 처리하는 기법을 제안하였다. 본 논문에서는 참고 문헌 [Kim01]의 명칭을 따라 이 기법을 Naive-Scan이라 부른다. 여과 단계 없이 모든 시퀀스들을 후보로 고려하는 Naive-Scan은 그 처리 성능이 떨어지므로, 참고 문헌 [Yi98]과 [Par00]에서는 여과 단계를 채택하는 기법들을 제안하였다. 본 논문에서는 참고 문헌 [Kim01]의 명칭을 따라 이 기법들을 각각 LB-Scan과 ST-Filter라 부른다. LB-Scan[Yi98]은 별도의 자료 구조 없이 모든 시퀀스들을 대상으로 여과 단계를 신속하게 수행하고, 이 결과 반환되는 후보 시퀀스들만을 대상으로 후처리 단계를 수행한다. 반면, ST-Filter[Par00]는 접미어 트리(suffix tree)라는 별도의 자료 구조를 이용하여 여과 단계를 수행하고, 이 결과 반환되는 후보 시퀀스들만을 대상으로 후처리 단계를 수행한다. LB-Scan과 ST-Filter 모두 후처리 단계에서는 Naive-Scan에서와 같이 동적 프로그래밍을 이용하여 타임 워핑 거리를 계산한다.

시퀀스 매칭은 다음과 같이 전체 매칭(whole matching)과 서브시퀀스 매칭(subsequence matching)으로 구분된다[Fal94].

- 전체 매칭: 시퀀스 S_1, S_2, ..., S_N을 포함하는 데이터베이스 D로부터 질의 시퀀스 Q와 유사한 시퀀스 S_i를 검색한다.
- 서브시퀀스 매칭: 시퀀스 S_1, S_2, ..., S_N을 포함하는 데이터베이스 D로부터 질의 시퀀스 Q와 유사한 서브시퀀스 $S_i[j:k]$를 포함하는 시퀀스 S_i와 j, k 값을 검색한다. 여기서 j와 k는 시퀀스 S_i내에서 질의 시퀀스와 유사한 서브시퀀스가 시작하는 위치와 끝나는 위치를 의미한다.

서브시퀀스 매칭은 전체 매칭을 일반화한 것이므로 [Fal94][Yi98], 실제 응용 분야에서 널리 사용된다. 따라서 본 논문에서는 보다 실용적인 타임 워핑 하의 효과적인 **서브시퀀스 매칭 문제**를 다루고자 한다. 기존에 제안된 LB-Scan 및 ST-Filter를 이용하여 실제 상황에서 사용 가능한 정도의 서브시퀀스 매칭 성능을 얻을 수 있으나, 실제 데이터베이스 환경에서 이러한 기법들을 이용한 서브시퀀스 매칭은 아직도 수초에서 수십 초의 매우 긴 처리 시간을 요구한다[Par00]. 따라서 보다 빠른 처리를 위한 추가의 성능 개선 방안이 요구된다.

Naive-Scan의 기본 전략은 각 시퀀스를 디스크로부터 액세스 한 후, 그 시퀀스 내의 각 서브시퀀스에 대하여 동적 프로그래밍을 이용하여 질의 시퀀스와의 타임 워핑 거리를 계산하는 것이다. 이러한 Naive-Scan의 기본 전략은 기존의 모든 타임 워핑 하의 서브시퀀스 매칭 기법들의 후처리 단계에서 공통적으로 사용된다. 따라서 Naive-Scan의 처리 과정의 최적화는 기존의 여과 단계를 포함하는 기법들의 수행 시간을 단축시키는 효과를 가지므로 타임 워핑 하의 서브시퀀스 매칭 연구에서 매우 중요한 의미를 가진다. 본 논문에서는 이 점을 착안하여 Naive-Scan의 CPU 처리 과정을 최적화하기 위한 새로운 기법을 제안한다.

제안된 기법은 질의 시퀀스와 서브시퀀스들간의 타임 워핑 거리들을 계산하는 과정에서 발생하는 많은 중복 작업 및 불필요한 작업을 사전에 제거함으로써 CPU 처리 성능을 극대화할 수 있다. 제안된 기법이 착오 기각을 발생시키지 않음과 Naive-Scan을 처리하기 위한 최적의 기법임을 이론적으로 증명한다. 다양한 실험들을 통한 성능 평가에 의하여 제안된 기법이 기존의 기법들에 미치는 성능 개선 효과를 정량적으로 검증한다.

2. 성능 최적화 방안

본 장에서는 타임 워핑 하의 서브시퀀스 매칭의 CPU 처리 과정에 대한 최적화 방안에 관하여 논의한다.

2.1. 이론적 배경

Naive-Scan CPU 처리 과정의 최적화를 위하여 본 논문에서 제안하는 기법은 다음과 같은 이론적 배경을 기반으로 하고 있다.

정리 1:

두 시퀀스 S와 Q에 대하여, S내의 임의의 접두어(prefix) S[1:j]와 Q와의 타임 워핑 거리 $D_{tw}(S[1:j], Q)$는 $D_{tw}(S, Q)$를 계산하기 위한 과정의 부산물(by-product)로서 함께 계산된다.

증명: 참고 문헌 [Kim04] 참조. □

정리 2:[2)]

$$\forall j\ (1 \le j \le |Q|)\ D_{tw}(S[1:i], Q[1:j]) > \varepsilon \Rightarrow \forall k\ (k > i)\ D_{tw}(S[1:k], Q(1:|Q|]) > \varepsilon$$

증명: 참고 문헌 [Kim04] 참조. □

2.2. 제안하는 기법

본 절에서는 위의 이론적인 배경을 기반으로 Naive-Scan을 이용한 타임 워핑 하의 서브시퀀스 매칭에서 CPU 처리 과정을 최적화하는 기법을 제안한다.

기존의 Naive-Scan의 방식에서는 시퀀스 S의 각 서브시퀀스 S[i:j] ($1 \le i \le |S|$, $i \le j \le |S|$)에 대하여 각각 서로 다른 거리 축적 테이블(distance accumulate table)[Yi98][Par00]을 구성한다. 시퀀스 S = <1, 2, 5, 7, 9, 8>와 질의 시퀀스 Q = <1, 3, 5, 8>를 고려해 보자. 그림 2.1은 질의 시퀀스 Q와 시퀀스 S의 서브시퀀스 S[1:4], S[1:5], S[1:6]의 각 거리 축적 테이블을 구성하는 과정을 나타낸 것이다. 그림에 나타난 바와 같이 세 거리 축적 테이블들 간에는 많은 중복이 존재한다. 서브시퀀스 S[1:4]에 대한 거리 축적 테이블은 서브시퀀스 S[1:5] 및 S[1:6]을 위한 거리 축적 테이블의 1~4행에 완전히 포함되며, 서브시퀀스 S[1:5]를 위한 거리 축적 테이블 역시 서브시퀀스 S[1:6]을 위한 거리 축적 테이블에 완전히 포함된다. 이와 같이, 기존의 Naive-Scan의 방식대로 모든 가능한 서브시퀀스들에 대하여 거리 축적 테이블을 개별적으로 구성하는 경우, 매우 많은 계산 과정의 중복이 발생하게 된다.

7	11	3	5	4
5	5	1	3	6
2	1	3	4	10
1	0	4	6	13
	1	3	5	8

9	19	7	9	5
7	11	3	5	4
5	5	1	3	6
2	1	3	4	10
1	0	4	6	13
	1	3	5	8

8	26	10	12	5
9	19	7	9	5
7	11	3	5	4
5	5	1	3	6
2	1	3	4	10
1	0	4	6	13
	1	3	5	8

그림 2.1. 동일한 접미어의 접두어인 세 서브시퀀스들에 대한 거리 축적 테이블.

본 연구에서는 이러한 계산 과정의 중복을 제거하기 위한 방법으로서 시퀀스 S의 모든 서브시퀀스가 아닌 모든 접미어 S[i:|S|]에 대해서만 거리 축적 테이블을 구성하는 방식을 제안한다. 시퀀스 S내의 접미어가 아닌 서브시퀀스 S[i:j] ($1 \le i < |S|$, $i \le j < |S|$)는 S의 접미어(suffix)인 S[i:|S|]의 접두어(prefix)이다. 정리 1에 의하여 접미어가 아닌 서브시퀀스 S[i:j]와 질의 시퀀스 Q와의 타임 워핑 거리 $D_{tw}(S[i:j], Q)$는 $D_{tw}(S[i:|S|], Q)$의 계산하기 위한 과정의 부산물로서 함께 계산된다. 따라서 시퀀스 S의 각 접미어 S[i:|S|]에 대하여 하나의 거리 축적 테이블을 구성함으로써 시퀀스 S에 포함되는 (|S|-i)개의 접미어가 아닌 서브시퀀스들에 대한 질의 시퀀스 Q와의 타임 워핑 거리 $D_{tw}(S[i:j], Q)$를 모두 얻을 수 있으며, 이 결과 기존 방식에서 발생하던 (|S|-i)회의 테이블 구성 시

2) 참고 문헌 [Par00]에서도 이와 유사한 정리를 제시하고, 이러한 특성을 이용한 처리 방안을 ST-Filter에 적용한 바 있다. 그러나 Naive-Scan에는 이러한 방식을 적용하지 않았다[Par03].

간을 모두 제거할 수 있다. 본 논문에서는 이러한 전략을 ***테이블 공유 전략(table-sharing strategy)***이라 정의한다.

그림 2.1의 예에 테이블 공유 전략을 사용하게 되면, S의 접미어의 하나인 S[1:6]에 대한 거리 축적 테이블만을 구성함으로써 이의 접두어에 해당되는 서브시퀀스 S[1:1], S[1:2], S[1:3], S[1:4], S[1:5], S[1:6]의 질의 시퀀스와의 타임 워핑 거리를 모두 구하게 된다.

본 연구에서는 CPU 수행 시간의 최적화를 위하여 정리 2를 이용한 또 하나의 전략을 제안한다. 테이블 공유 전략에 의한 시퀀스 S의 접미어 S[i:|S|]에 대한 거리 축적 테이블 구성하는 것은 S[i:|S|]의 접두어인 (|S|-i+1)개의 서브시퀀스들 S[i:j]에 대한 질의 시퀀스 Q와의 타임 워핑 거리들을 j가 증가하는 순서로 구하는 과정을 의미한다. 정리 2는 이 과정에서 접미어 S[i:|S|]에 대한 거리 축적 테이블을 j가 |S|로 될 때까지 완전히 구성할 필요가 없음을 알려주는 것이다. 즉, 테이블 구성 과정에서 j 열에 존재하는 모든 값이 질의에서 주어진 ε보다 크다면 더 이상의 진행은 무의미하다는 것이다. 따라서 더 이상의 테이블 구성의 진행을 중단하고, 길이가 더 긴 서브시퀀스들을 최종 결과에서 안전하게 제외시킬 수 있다. 본 논문에서는 이 전략을 ***테이블 커팅 전략(table-cutting strategy)***이라 정의한다.

그림 2.1에서 가장 오른쪽 그림을 S의 접미어 S[1:6]에 대한 거리 축적 테이블만을 구성하는 과정이라 가정하자. 질의에서 주어진 ε이 2라 하면, S[1:6]의 접두어에 해당되는 네 번째 서브시퀀스 S[1:4]와 대응되는 행 내의 모든 값이 2보다 크므로 테이블 구성은 중단된다. 따라서 이후의 행과 대응되는 서브시퀀스들에 대한 타임 워핑 거리 계산 과정을 제거시킬 수 있다.

정리 3 :
제안된 기법을 이용하여 타임 워핑 하의 서브시퀀스 매칭을 수행하는 경우 착오 기각이 발생하지 않는다.

증명: 참고 문헌 [Kim04] 참조. □

다음은 제안된 기법의 효율성을 규명한다. 기존의 Naive-Scan 방식에서는 모든 서브시퀀스들을 위한 별도의 거리 축적 테이블을 모두 구성한다. 시퀀스 S내에는 모두 $O(|S|^2)$ 개의 서브시퀀스들이 존재한다. 각 서브시퀀스 S[i:j]와 질의 시퀀스 Q간의 거리 축적 테이블을 구하는 시간은 $O(|Q|*|S[i:j]|)$이다. 따라서 질의 시퀀스 Q와 시퀀스 S에 대한 서브시퀀스 매칭을 수행하기 위한 CPU 처리 시간은 $O(|Q|*|S|^3)$이다.

반면, 제안된 기법에서는 테이블 공유 전략에 의하여 시퀀스 S내의 접미어들에 대해서만 거리 축적 테이블을 구성하게 된다. 시퀀스 S내에는 $O(|S|)$개의 접미어들이 존재하므로, 제안된 기법으로 질의 시퀀스 Q와 시퀀스 S에 대한 서브시퀀스 매칭을 수행하기 위한 CPU 처리 시간은 $O(|Q|*|S|^2)$가 된다. 따라서 기존의 타임 워핑 하의 서브시퀀스 매칭의 수행 시간을 크게 줄일 수 있다. 뿐만 아니라, 테이블 커팅 전략에 의하여 더 이상의 진행이 무의미한 경우에는 거리 축적 테이블을 구성을 도중에 중단하게 되므로 수행 시간은 더욱 줄어든다.

제안된 기법은 기존의 여과 단계를 포함하는 방식인 LB-Scan과 ST-Filter의 후처리 단계에도 성공적으로 적용될 수 있다.

3. 성능 평가

본 장에서는 실험에 의한 성능 분석을 통하여 제안된 기법을 채택함으로써 나타나는 성능 개선 효과를 정량적으로 규명한다.

3.1 실험 환경

본 연구에서는 성능 분석을 위하여 실제 데이터베이스 K_Stock_Data를 사용하였다. K_Stock_Data는 한국의 실제 주식 데이터로서 길이가 300인 620개의 데이터 시퀀스들로 구성된다.

질의 시퀀스 Q는 데이터베이스로부터 임의로 선택한 한 시퀀스로부터 길이가 Len(Q)인 임의의 위치의 서브시퀀스를 선택하여 사용하는 방법으로 생성하였다. 질의 구성 시에는 질의 선택률(query selectivity)을 아래의 식과 같이 정의하고, 각 질의에 대하여 원하는 선택률을 만족하도록 허용치 ε을 조정하였다.

$$\text{선택률} = \frac{\text{질의 시퀀스 } Q\text{와 } \varepsilon - \text{매치하는 모든 서브시퀀스들의 수}}{\text{길이가 } \mathrm{Len}(Q)\text{인 모든 데이터 서브시퀀스들의 수}}$$

실험을 위한 하드웨어 플랫폼은 1.7 GHz Pentium IV와 1,280 MB의 주기억장치가 장착된 PC를 사용하였으며, 소프트웨어 플랫폼은 운영 체제 Linux Kernel Version 2.4.18 및 컴파일러 Glibc 2.2.4를 사용하였다.

3.2. 실험 결과 및 분석

실험 1의 목적은 선택률의 변화에 따르는 제안된 최적화 기법의 성능 개선 효과를 규명하기 위한 것이다. 질의 선택률을 변화시키면서 제안된 최적화 기법과 기존 기법을 각각 후처리 단계에서 채택하는 Naive-Scan, LB-Scan, ST-Filter의 성능을 비교하였다. 사용된 질의 선택률은 7.10×10^{-8}(최종 결과: 2개), 2.13×10^{-7}(최종 결과: 6개), 3.55×10^{-7}(최종 결과: 10개), 4.97×10^{-7}(최종 결과: 14개)이다. 사용된 질의 시퀀스의 길이는 110이다.

그림 3.1은 전체 수행 시간을 표현한 것이다. 가로축은 선택률의 크기를 나타내며, 세로축은 타임 워핑 하의 서브시퀀스 매칭의 전체 처리 시간을 나타낸다. 처리 시간이 로그 스케일로 표현되었음에 유의해야 한다. ST-Filter의 경우, 최적화의 성능 개선 효과가 미미하므로, 최적화 전과 후의 그래프가 거의 동일한 형태로 나타났다. LB-Scan의 경우, 최적화에 의하여 상당한 성능 개선의 효과를 보였다. Naive-Scan의 경우, 최적화로 인한 성능 개선 효과가 가장 두드러지게 나타났다. 특히, Naive-Scan이 후처리 단계의 최적화 전에는 가장 떨어지는 성능을 보였으나, 최적화 후에는 선택률의 변화에 관계없이 모든 경우에서 ST-Filter나 LB-Scan을 사용한 경우보다 훨씬 더 좋은 성능을 보이는 것으로 나타났다. 이러한 결과는 성능 병목인 CPU 처리 과정을 최적화함으로써 기존 기법들인 Naive-Scan, LB-Scan, ST-Filter 간의 처리 성능 상의 순위 역전 현상이 발생하였음을 보이는 것이다. 이러한 결과는 기존에 전혀 예상하지 못한 매우 놀라운 현상이다.

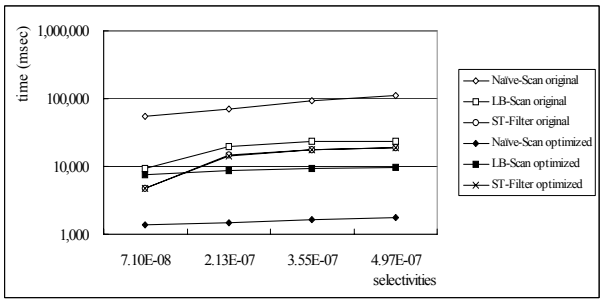

그림 3.1. 선택률 변화에 따른 성능 결과.

실험 2에서는 최대 WarpRatio[Ber96]를 변화시키면서 제안된 최적화 기법과 기존 기법을 각각 후처리 단계에서 채택하는 Naive-Scan, LB-Scan, ST-Filter에 대한 실험을 수행하였다. 사용된 최대 WarpRatio는 3, 6, 9, 12이다. 실험 2의 목적은 최대 WarpRatio의 변화에 따라 제안된 최적화 기법의 성능 개선 효과가 어떻게 나타나는가를 규명하기 위한 것이다. 사용된 질의 시퀀스의 길이는 110이며, 선택률은 2.13×10^{-7}을 사용하였다.

그림 3.2는 실험 결과를 나타낸 것이다. 가로축은 최대 WarpRatio의 크기를 나타내며, 세로축은 타임 워핑 하의 서브시퀀스 매칭의 전체 수행 시간을 로그 스케일로 나타낸 것이다. 최대 WarpRatio가 증가함에 따라 모든 경우에서 전체 수행 시간은 점차 증가하는 것으로 나타났다. 이것은 최대 WarpRatio가 클수록 서브시퀀스 내의 각 요소 값이 반복될

수 있는 회수가 다양해지므로, CPU 처리 시간이 증가되기 때문이다. 실험 1의 결과와 마찬가지로, 제안된 최적화 기법은 ST-Filter를 제외한 LB-Scan 및 Naive-Scan에서 큰 성능 개선 효과를 보였다. 특히, 최적화로 인한 Naive-Scan의 성능 향상 효과가 42배에서 52배로 가장 큰 것으로 나타났으며, 이 결과 전체 수행 시간 측면에서 다른 모든 기법들과 비교하여 월등한 성능을 보였다. 이 최적화된 Naive-Scan은 두 번째로 좋은 성능을 보인 최적화된 LB-Scan에 비해 5배에서 6배까지 더 좋은 성능을 보였다.

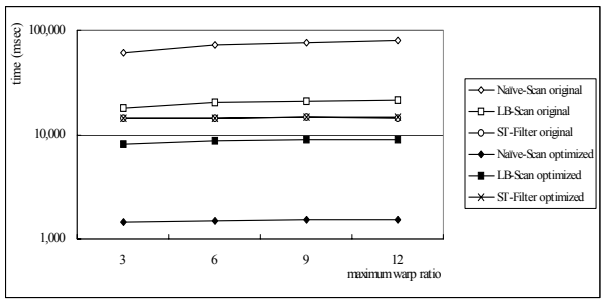

그림 3.2. 최대 WarpRatio 변화에 따른 성능 결과.

실험 3에서는 다양한 길이의 질의 시퀀스에 대하여 제안된 최적화 기법과 기존 기법을 각각 후처리 단계에서 채택하는 Naive-Scan, LB-Scan, ST-Filter에 대한 실험을 수행하였다. 사용된 질의 시퀀스의 길이는 80, 140, 200, 260이다. 실험 3의 목적은 질의 시퀀스 길이의 변화에 따르는 제안된 최적화 기법의 성능 개선 효과를 규명하기 위한 것이다. 사용된 최대 WarpRatio는 5이며, 선택률은 2.13×10^{-7}을 사용하였다.

그림 3.3은 실험 결과를 나타낸 것이다. 가로축은 사용된 질의 시퀀스 길이를 나타내며, 세로축은 타임 워핑 하의 서브시퀀스 매칭의 전체 수행 시간을 로그 스케일로 나타낸 것이다. 질의 시퀀스의 길이가 길어짐에 따라 모든 경우에서 전체 수행 시간은 크게 증가하는 것으로 나타났다. 이는 서브시퀀스 S와 질의 시퀀스 Q에 대한 거리 축적 테이블을 구성하는 시간이 O(|S|×|Q|)로 나타나기 때문이다. 전체적인 경향은 이전의 다른 실험들에서와 마찬가지로 나타났으며, Naive-Scan에서의 제안된 최적화 기법의 성능 향상 효과가 가장 크고, 이 결과 최적화된 Naive-Scan이 가장 좋은 처리 성능을 보였다.

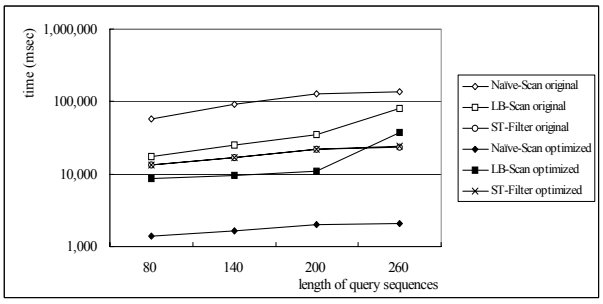

그림 3.3. 질의 시퀀스 길이의 변화에 따른 성능 결과.

4. 결론

본 논문에서는 시계열 데이터베이스에서 타임 워핑 하의 서브시퀀스 매칭을 효과적으로 처리하는 방안에 대하여 논의하였다. 본 논문의 주요 공헌은 다음과 같이 요약될 수 있다.

- 기존의 가장 기본적인 처리 방식인 Naive-Scan의 성능 병목이 CPU 처리에 있음을 보이고, CPU 처리 과정의 최적화가 Naive-Scan의 전체 성능을 개선할 수 있는 매우 중요한 이슈임을 지적하였다.
- Naive-Scan의 CPU 처리 과정을 최적화하는 새로운 기법을 제안하였다. 제안된 기법은 질의 시퀀스와 서브시퀀스들간의 타임 워핑 거리들을 계산하는 과정에서 발생하는 많은 중복 작업을 사전에 제거함으로써 CPU 처리 성능을 극대화할 수 있다.
- 제안된 기법이 착오 기각을 발생시키지 않음과 Naive-Scan을 처리하기 위한 최적의 기법임을 보였다.
- 제안된 기법은 기존의 각 타임 워핑 하의 서브시퀀스 매칭 기법인 LB-Scan과 ST-Filter의 후처리 단계에 적용하는 것이 가능하다.
- 다양한 실험을 통한 성능 평가에 의하여 제안된 후처리 단계의 최적화 기법이 가져오는 성능 개선 효과를 정량적으로 검증하였다.

실험 결과에 의하면, 기존의 타임 워핑 하의 서브시퀀스 매칭을 위한 모든 기법들이 제안된 최적화 기법에 의하여 성능이 개선되는 것으로 나타났다. 특히, 가장 기본적인 기법인 Naive-Scan은 제안된 최적화 기법의 적용 전에는 가장 떨어지는 성능을 보였으나, 최적화 기법의 적용 후에는 모든 경우에서 ST-Filter나 LB-Scan을 사용한 경우보다 더 좋은 성능을 보였다. 이러한 결과는 기존에 전혀 예상하지 못한 매우 놀라운 것이며, 성능 병목인 CPU 처리 과정을 최적화함으로써 기존 기법들인 Naive-Scan, LB-Scan, ST-Filter 간의 처리 성능 상의 순위 역전 현상이 발생하였음을 보이는 것이다.

최적화된 Naive-Scan의 성능의 우수성은 (1) 시퀀스 길이가 길수록, (2) 저장된 시퀀스 개수가 많을수록, (3) 선택률이 작을수록 더 두드러지는 것으로 나타났다. 이러한 경향은 실제 대형 데이터베이스 환경의 특성을 고려할 때 매우 바람직한 것이다.

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.

[Ber96] D. J. Berndt and J. Clifford, "Finding Patterns in Time Series: A Dynamic Programming Approach," *Advances in Knowledge Discovery and Data Mining*, pp. 229-248, 1996.

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

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

[Kim04] S. W. Kim et al., Subsequence Matching Under Time Warping in Time-Series Databases: Observation, Optimization, and Performance Results, 2004. (unpublished manuscript)

[Par00] S. H. Park et al., "Efficient Searches for Similar Subsequences of Difference Lengths in Sequence Databases," In *Proc. Int'l. Conf. on Data Engineering*, IEEE ICDE, pp. 23-32, 2000.

[Par01] S. H. Park, S. W. Kim, J. S. Cho, and S. Padmanabhan, "Prefix-Querying: An Approach for Effective Subsequence Matching Under Time Warping in Sequence Databases," In *Proc. ACM Intl. Conf. on Information and Knowledge Management*, ACM CIKM, pp. 255-262, 2001.

[Par03] S. H. Park, *private communication*, 2003.

[Rab93] L. Rabiner and H. H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[Yi98] B. K. Yi, H. V. Jagadish, and C. Faloutsos, "Efficient Retrieval of Similar Time Sequences Under Time Warping," In *Proc. Int'l. Conf. on Data Engineering*, IEEE ICDE, pp. 201-208, 1998.