제22회 한국정보처리학회 추계학술발표대회 논문집 제11권 제2호 (2004. 11)

유사 멜로디 검색을 이용한 음악 표절 감지 시스템

박정일*, 김상욱**
*강원대학교 컴퓨터정보통신학부
**한양대학교 정보통신학부

A Music Plagiarism Detection System Using Similar Melody Searching

Jeong-Il Park* and Sang-Wook Kim**
*Dept. of Comp., Info., & Comm. Eng., Kangwon National University
**College of Info. & Comm. Hanyang University

요 약

유사 멜로디 검색은 질의 멜로디와 유사한 멜로디들을 음악 데이터베이스로부터 찾는 연산이다. 본 논문에서는 유사 멜로디 검색을 기반으로 하는 표절 감지 시스템 개발에 관하여 논의한다. 먼저, 정합 및 이동 변환을 지원하는 새로운 유사 모델을 제안한다. 또한, 각 멜로디의 특징들을 인덱싱 하는 방법과 인덱스를 기반으로 표절 감지를 처리하는 방법을 제시한다. 제안된 표절 감지 시스템을 이용하여 작곡가는 자신의 멜로디와 유사한 멜로디를 가지는 곡들을 음악 데이터베이스에서 효과적으로 검색할 수 있다. 실험을 통한 성능 평가를 통하여 제안된 기법의 우수성을 규명한다. 실험 결과에 의하면, 제안된 기법은 순차 검색을 기반으로 하는 방법과 비교하여 약 31배까지의 성능 개선 효과를 보였다.

1. 서론

최근, 이미지, 비디오, 오디오 등 다양한 멀티미디어 데이터의 사용이 폭발적으로 증가함에 따라 이를 위한 내용 기반 검색에 많은 관심이 기울여지고 있다[12][7][13][5][4][3]. 내용 기반 검색은 크게 인덱싱 기술과 질의 처리 기술로 분류되며, 이러한 두 가지 기술들은 기존의 연구 결과에 의하여 큰 발전을 이루었다. 그러나 이미지나 비디오에 관한 검색 기술들에 비하여 오디오에 대한 검색 기술은 그 발전이 상대적으로 적은 것으로 알려져 있다[8].

유사 멜로디 검색(similar melody searching)은 질의 멜로디와 유사한 멜로디들을 음악 데이터베이스로부터 찾는 연산이다. 본 연구에서는 유사 멜로디 검색을 기반으로 하는 표절 감지 시스템 개발에 관하여 논의한다. 표절 감지 시스템은 기 작곡된 곡들이 저장된 데이터베이스 내에 작곡가가 새롭게 작곡한 멜로디와 유사한 멜로디가 존재하는가를 점검해 주는 시스템이다. 선의의 작곡가도 자신도 모르는 사이 본의 아닌 표절 시비에 휘말릴 수 있다. 본 연구의 목적은 이 표절 감지 시스템을 이용하여 유사한 멜로디를 가지는 기존의 곡이 존재하는가를 점검함으로써 작곡가가 불필요한 표절 시비를 사전에 방지할 수 있도록 하기 위한 것이다.

기존 연구에서 제안된 시스템들[6][8][9][10]과 비교하여 본 연구에서 제안하는 표절 감지 시스템의 주요 특성들은 다음과 같다. 이러한 특성들은 본 연구에서 제안한 새로운 기법들로 인한 것이다.

- 새로운 유사 모델: 정합 및 이동 변환을 유사 모델 내에 채택함으로써 실제 사용자들이 유사하다고 판정하는 곡들을 시스템이 유사하지 않다고 판정하는 문제점을 해결한다.
- 다차원 인덱스의 채택: 멜로디의 특징들을 대상으로 다차원 인덱스를 구축함으로써 빠른 검색의 기반을 마련한다.
- 삼 단계 질의 처리: 인덱스 검색, 윈도우 스티칭, 후처리의 삼 단계로 구성되는 질의 처리 방식을 채택함으로써 빠른 검색 성능을 제공한다.

실험을 통한 성능 평가를 통하여 제안된 기법의 우수성을 규명한다. 실험 결과에 의하면, 제안된 기법은 순차 검색을 기반으로 하는 방법과 비교하여 약 31배까지의 성능 개선 효과를 보이는 것으로 나타났다.

본 논문의 구성은 다음과 같다. 제 2장에서는 서로 다른 두 멜로디의 유사한 정도를 판정하기 위한 유사 모델을 제안한다. 제 3장에서는 유사 멜로디를 효과적으로 검색하기 위한 멜로디 데이터베이스 인덱싱 방안에 관하여 논의한다. 제 4장에서는 제안된 인덱싱 방안을 기반으로 하는 삼 단계 질의 처리 기법을 제시한다. 제 5장에서는 제안하는 기법의 우수성을 규명하기 위한 성능 평가 결과를 제시한다. 끝으로, 제 6장에서는 본 논문을 요약하고, 결론을 내린다.

2. 유사모델

본 장에서는 서로 다른 두 멜로디들이 얼마나 유사한가를 판정하기 위한 유사 모델을 제시한다.

2.1. 기본 모델

곡 내의 한 멜로디(melody)는 마디(snatch)들의 리스트로 정의되며, 다음과 같은 시퀀스 $S = <(s_i, sL_i)> (0<=i<n)$로 표현할 수 있다. 본 논문에서는 이를 멜로디 시퀀스(melody sequence)라 정의한다. 여기서, s_i는 멜로디 내 i번째 음의 높이를 의미하며, sL_i는 멜로디 내 i번째 음의 길이를 의미한다. 또한, 음의 수에 해당하는 n을 멜로디 시퀀스의 길이라 부른다. 예를 들어, 4/4박자의 다장조 곡에 포함되는 아래 멜로디는 하나의 마디로 구성되며, 길이가 4인 멜로디 시퀀스 <(솔, 1), (라, 1/2), (라, 1/2), (파, 2)>로 표현할 수 있다[1].

먼저, 임의의 두 멜로디 시퀀스 $S = <(s_i, sL_i)>(0<=i<n)$와 $Q = <(q_j, qL_j)>(0<=j<m)$를 비교할 수 있는 전제 조건은 다음과 같다.

- 전제 조건 1: S와 Q는 동일한 박자를 갖는다. 즉, 서로 다른 박자를 갖는 두 멜로디 시퀀스들은 유사 짐검 대상에서 사전에 제외된다.
- 전제 조건 2: n=m 이고, 모든 (i, j)에 대하여 i=j 이면, sL_i =

1) 본 논문에서는 설명의 편의상 이와 같이 음 높이는 실수로 음 길이는 분수로 표현한 것이다. 그러나 실제로 미디 표준 파일(standard MIDI file)[11]에 저장될 때, 음 높이와 음 길이는 각각 정수로 표현된다.

qL_j 이다. 즉, 두 멜로디 시퀀스들의 길이가 동일해야 함을 의미하며, 대응되는 각 음의 길이도 동일해야 함을 의미한다.

전제 1과 전제 2를 모두 만족하는 두 멜로디 시퀀스 S와 Q가 유사한지의 여부를 판정하는 기준으로 아래 정의 1을 이용한다.

정의 1: 두 멜로디 시퀀스 $S = <(s_i, sL_i)>(0<=i<n)$와 $Q = <(q_j, qL_j)>(0<=j<n)$는 위의 전제 조건 1, 2와 아래 조건을 모두 만족할 때, 서로 유사하다고 판정한다.

$$L_\infty(S,Q) = L_\infty(<s_0, s_1,..., s_{n-1}>, <q_0, q_1, ..., q_{n-1}>) < \varepsilon$$

여기서, ε은 사용자에 의하여 제시된 유사 허용치이다. 또한, L_∞는 대응되는 두 음간의 차들 중 최대 값을 반환하는 함수이다[1].

이 정의는 두 멜로디 내에 대응되는 두 음의 높이 차 중 최대인 것이 허용치 ε 이하인 경우에 한하여 두 멜로디를 유사하다고 간주함을 의미한다. 즉, 두 멜로디 S와 Q가 유사하기 위한 조건은 멜로디 S의 모든 음의 높이는 항상 또 다른 멜로디 Q내 대응되는 음의 높이와 비교하여 일정 범위 ε내에 존재해야 함을 의미한다.

2.2. 정합 연산

제 2.1절에서 언급한 기본 유사 모델에서는 두 멜로디 시퀀스의 길이가 동일하고, 대응되는 각 음의 길이도 동일해야 함을 전제로 유사 정도를 측정한다. 그러나 이러한 전제는 응용에서 실제로 유사한 멜로디들을 비교 대상에서 제외하는 문제점을 갖는다. 아래 그림 1을 예로 살펴보자. 두 멜로디는 거의 동일한 것이지만, 전제 조건 2에 의하여 유사 비교 대상에서 사전에 제외된다.

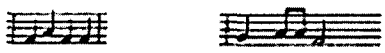

그림 1. 서로 다른 길이를 갖는 두 멜로디.

본 연구에서는 이러한 문제점을 해결하기 위하여 정합 연산(alignment)을 도입한다. 정합 연산은 비교 대상인 두 멜로디 시퀀스의 길이와 각각 대응되는 음의 길이를 모두 같도록 만드는 과정이다. 단, 이러한 정합은 동일한 박자와 동일한 마디 수를 갖는 두 멜로디에 대해서만 적용이 가능하다.

정합은 음의 분할(split) 연산을 기반으로 한다. 분할은 비교 대상이 되는 한 음의 길이가 대응시키고자 하는 다른 음의 길이보다 긴 경우, 이를 동일한 높이의 다수의 음들로 나누는 역할을 한다. 그림 2는 대응되는 음의 길이가 서로 다른 두 멜로디에 대하여 음 분할을 기반으로 하는 정합을 수행한 결과를 보여준다. 이와 같은 정합 연산의 사용으로 동일한 박자와 동일한 마디를 갖는 모든 멜로디들의 유사 비교가 가능해 진다[2]. 이후부터는 비교의 대상이 되는 두 멜로디 S와 Q가 동일한 길이를 가지며, 각각 대응되는 음이 동일한 길이를 갖도록 정합 연산을 적용하였다는 전제 하에 논의를 진개한다.

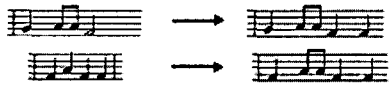

그림 2. 정합 연산의 적용.

2.3. 이동 변환

제 2.1절에서 언급한 기본 유사 모델에서는 음의 절대 높이만을 기준으로 유사 정도를 측정하므로 절대 높이가 서로 다른 유사한 두 멜로디를 유사하지 않는 것으로 판정하는 문제를 갖는다. 예를 들어, 두 멜로디 A와 B는 대응되는 음 높이의 변화와 길이는 완전히 일치하고, 단지 절대 음 높이에 있어서만 차이가 있다고 하자. 사람들은 이 두 멜로디가 매우 유사하다고 느끼지만, 위의 기본 유사 모델을 사용하는 경우 두 멜로디는 유사하지 않다고 판정된다.

본 연구에서는 이 문제를 해결하기 위하여 이동 변환(shift)을 사용한다. 이동 변환은 아래의 식을 이용하여 멜로디 시퀀스 $S(=<(s_i, sL_i)>, (0<=i<n))$를 S'으로 변환한다.

$$SHIFT(S) = S' = <(s_i', sL_i)> \ (0<=i<n),\ s_i' = s_i - (max_s + min_s)/2$$

여기서, max_s와 min_s는 각각 멜로디 시퀀스 S의 음들 중 최대 높이와 최소 높이를 의미한다.

이와 같이, 이동 변환을 채택함으로써 본 연구에서 제안하는 유사 모델은 음 높이의 변화가 매우 유사하고 단지 절대 음 높이에 있어서만 차이가 있는 두 멜로디의 유사성을 올바르게 판정할 수 있다. 따라서 두 멜로디 시퀀스 S와 Q가 유사한지의 여부를 판정하는 기준은 정의 2와 같이 재 정의된다.

정의 2: 두 멜로디 시퀀스 $S = <(s_i, sL_i)>(0<=i<n)$와 $Q = <(q_j, qL_j)>(0<=j<n)$는 아래 조건을 만족할 때, 서로 유사하다고 판정한다.

$$L_\infty(SHIFT(S), SHIFT(Q)) = L_\infty(<s_0', s_1', ..., s_{n-1}'>, <q_0', q_1', ..., q_{n-1}'>) < \varepsilon$$

여기서, $s_i' = s_i - (max_s + min_s)/2$, $q_i' = q_i - (max_q + min_q)/2$ 이다. 또한, ε은 사용자에 의하여 제시된 유사 허용치이며, L_∞는 대응되는 두 음간의 차들 중 최대 값을 반환하는 함수이다.

이와 같은 이동 변환을 이용한 유사 모델을 통하여 유사 멜로디 검색에서는 두 멜로디의 절대 음 높이를 무시하도록 하고, 음의 변화에 초점을 맞추어 두 멜로디간의 유사 정도를 측정하게 된다.[3]

3. 인덱싱 방안

본 장에서는 질의 멜로디와 유사한 멜로디들을 음악 데이터베이스로부터 효과적으로 검색하기 위한 인덱싱 방안에 관하여 논의한다.

3.1. 특징의 추출

먼저, 한 멜로디 내의 k개의 연속된 마디들을 윈도우(window)라 정의한다. 본 연구에서는 k-마디들로 구성되는 윈도우를 인덱싱의 단위로 채택한다. 즉, 각 마디로부터 특징들을 추출하고, k개의 연속된 마디들의 특징들을 묶어 이를 하나의 단위로 인덱싱을 수행한다. 멜로디의 각 마디로부터 추출하는 특징들은 timeMeter, maxPitch, minPitch 이다. 여기서, timeMeter는 해당 마디의 박자를 나타내며, maxPitch와 minPitch는 해당 마디 내의 최고 음 높이와 최저 음의 높이를 의미한다.

3.2. 인덱스 구조

제 3.1절에서 제시한 인덱싱 특징들은 다수이므로 본 연구에서는 이들을 인덱싱하기 위한 구조로서 R*-트리[2]를 사용한다. R*-트리는 가장 널리 사용되는 다차원 인덱스 구조(multidimensional index structure)로서 다수의 특징들을 효과적으로 인덱싱하기 위하여 사용된다.

그림 3은 제안하는 기법에서 사용하는 R*-트리의 단말 노드 엔트리의 구조를 나타낸 것이다. 여기서는 편의상 윈도우를 구성하는 마디 수 k가 4인 경우를 가정하여 설명한다. 즉, 윈도우 내 각 마디에 대한 <최고 음 높이, 최저 음 높이>의 쌍들과 해당 곡의 박자, 해당 윈도우를 포함하는 곡의 식별자, 이 곡에서의 해당 윈도우의 시작 위치 등이 기록된다. songID는 전체 데이터베이스에서 해당 곡을 유일하게 식별할 수 있는 식별자이며, snatchNo는 해당 곡 내에서 이 마디의 위치를 식별할 수 있는 식별자이다.

max Pitch 1	min Pitch 1	max Pitch 2	min Pitch 2	max Pitch 3	min Pitch 3	max Pitch 4	min Pitch 4	time Meter	song ID	snatch No

그림 3. 단말 노드 엔트리의 구조.

본 연구에서는 그림 3에 나타난 애트리뷰트들 중, R*-트리를 위한 구성 애트리뷰트(organizing attribute)[14]를 다음과 같이 결정한다. 먼저, songID와 snatchNo는 검색 조건을 위한 애트리뷰트로 사용되지 않으므로 구성 애트리뷰트에서 제외한다. 각 maxPitch i와 minPitch i는 하나의 값인 pitchRange i(= maxPitch i - minPitch i)로 표현할 수 있다. 이러한 경우, 구성 애트리뷰트의 수를 k*2 차원에서 k차원으로 줄일 수 있으므로, k개의 maxPitch i와 minPitch i 대신 k 개의 pitchRange i를 구성 애트리뷰트로 사용하는 방식을 채택한다. 또한, timeMeter를 이

---

2) 몇몇 기존 연구에서 분할 연산과 반대되는 개념인 병합(merge) 연산을 분할 연산과 함께 사용하는 방법을 제시하였다[9]. 그러나 분할 연산만을 이용한 정합도 모든 경우에 적용 가능하다.

3) 이러한 유사 모델은 요소 값의 크기가 서로 크게 다른 두 시퀀스의 유사성을 비교하기 위하여 실제 응용에서 널리 사용하는 방식이다. 이러한 기존의 응용에서는 스케일링 변환과 이동 변환을 통합한 형태인 정규화(normalization) 변환을 주로 사용하지만, 두 멜로디들 간의 유사성 측정에서는 스케일링 변환은 적합하지 않으므로, 본 연구에서는 이동 변환만을 채택한 것이다.

와는 독립적인 k+1번째 구성 애트리뷰트로 채택한다. 따라서 각 윈도우는 k+1 차원 공간 상의 하나의 점으로 사상된다. 본 연구에서는 이를 데이터 윈도우 점이라 부른다. 그림 4는 R*-트리의 비단말 노드 엔트리의 구조를 나타낸 것이다. 마지막 필드인 nextLevelNode는 R*-트리 상의 다음 단계 노드를 가리키는 포인터 역할을 한다.

R*-트리는 이러한 k+1차원 공간상에 존재하는 다수의 데이터 윈도우 점들을 저장 및 관리하게 된다. 따라서 R*-트리 검색 단계에서는 이러한 다수의 데이터 윈도우 점들 중 최종 질의 결과에 포함될 가능성이 높은 후보 윈도우 점들을 반환한다. 반면, 그림 3에서 제시된 maxPitch i와 minPitch i 등은 R*-트리 검색에는 직접 사용되지 않으며, 제 4장에서 설명할 윈도우 스티칭 단계에서 후보 윈도우 점들의 수를 더욱 줄이기 위하여 사용된다.

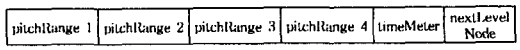

그림 4. 비단말 노드 엔트리의 구조.

3.3. 다차원 인덱스의 구성

데이터베이스 내에 저장된 전체 곡들에 대하여 R*-트리를 구성하는 절차는 아래 알고리즘 1과 같다. 여기서, 슬라이딩 윈도우(sliding window)는 해당 곡의 모든 가능한 위치의 마디에서 추출한 윈도우를 의미한다. 반면, 제 4장에서 언급하는 디스조인트 윈도우(disjoint window)는 해당 곡에서 서로 다른 윈도우가 전혀 겹치지 않도록 하는 방식으로 추출한 윈도우를 의미한다.

> FOR each song accessed from a music database,
> 1. extract *sliding windows*, each of which consists of k pitches, from the song;
> 2. FOR each sliding window,
> 2.1. obtain F_{sw}, the features mentioned in Figures 3.1 and 3.2, from the sliding window;
> 2.2. insert F_{sw} *into the* R*-tree;

알고리즘 1. R*-트리의 구성 절차.

4. 질의 처리 방안

본 장에서는 제 3장에서 제안한 인덱스를 기반으로 유사 멜로디 검색을 효과적으로 처리하는 질의 처리 방안에 관하여 설명한다. 유사 멜로디 검색 질의에서 주어지는 멜로디를 질의 멜로디라 정의한다. 전체 질의 처리는 질의 멜로디 특징 추출, R*-트리 검색, 윈도우 스티칭, 후처리의 순으로 진행된다. 본 장에서는 이러한 각 단계에 대하여 상세히 설명한다.

4.1. 질의 멜로디 특징 추출

먼저, 질의 멜로디로부터 k 마디로 구성된 디스조인트 윈도우(disjoint window) dw들을 추출한다. 디스조인트 윈도우들은 서로 공통되는 마디가 발생하지 않는 방식으로 추출된 윈도우들을 의미한다. 각 dw로부터 k개의 연속된 마디 내의 최대 음 높이(maxPitch)와 최소 음 높이(minPitch), 그리고 해당 dw의 박자 정보(timeMeter)를 윈도우 스티칭을 위한 특징으로서 구한다. 또한, maxPitch i와 minPitch i로부터 pitchRange i를 구함으로써 각 dw를 하나의 k+1차원 공간상의 하나의 점으로 표현한다. 본 연구에서는 이를 질의 윈도우 점이라 부른다.

각 질의 윈도우 점에 대하여 질의 윈도우 점을 확장하여 중심으로 각 변의 길이까지의 거리가 ε인 k 차원 정사각형을 구한다. 단, 마지막 차원인 박자 정보에 대한 차원은 ε만큼의 확장을 하지 않는다. 그 이유는 유사 멜로디 검색에서 음의 폭에 대해서는 ε만큼의 유사 허용치를 부여하지만, 박자에 대해서는 완전히 일치하는 곡을 찾아야 하기 때문이다. 여기서, ε은 질의에 주어진 유사 허용치를 의미한다. 본 연구에서는 이 확장된 k 차원 정사각형을 질의 사각형이라 부른다.

4.2. 인덱스 검색

인덱스 검색은 질의 멜로디와 유사할 가능성이 높은 후보 멜로디들을 찾기 위한 제 1차 필터링 과정이다. R*-트리 인덱스를 탐사함으로써 각 질의 직사각형 내에 포함되는 데이터 윈도우 점들을 찾는다. 본 연구에서는 이들을 후보 데이터 윈도우 점이라 부른다. 그림 5는 다차원 공간상의 데이터 윈도우 점들과 질의 윈도우 점들의 분포를 나타낸 것이다. 도면화의 편의상, 이차원 공간상의 분포를 가정한다. 여기서, X로 표현된 점이 질의 윈도우 점이며, 나머지 점들은 데이터 윈도우 점들을 의미한다. 또한, 사각형은 질의 사각형을 나타내며, 이 내부에 들어온 점들이 인덱스 검색 결과 반환되는 후보 데이터 윈도우 점들을 나타낸다.

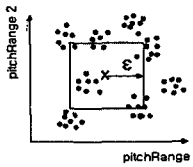

그림 5. 질의 사각형과 데이터 윈도우 점.

4.3. 윈도우 스티칭

윈도우 스티칭은 질의 멜로디와 유사할 가능성이 높은 후보 멜로디들을 찾기 위한 제 2차 필터링 과정이다. 인덱스 검색의 결과로 반환되는 후보 데이터 윈도우들 중에서 질의 멜로디와 유사할 가능성이 없는 데이터 멜로디에 속하는 것들을 제거한다. 윈도우 스티칭의 기본 아이디어는 각 질의 윈도우와 유사하다고 판단되어 검색된 후보 윈도우 집합들로부터 모든 질의 윈도우들과 매치되는 연속된 데이터 윈도우들을 포함하는 멜로디들만을 골라내는 것이다.

그림 6은 윈도우 스티칭 과정을 예로 나타낸 것이다. 먼저, 위의 세 사각형들은 질의 멜로디로부터 추출된 세 개의 디스조인트 윈도우들을 나타낸다. 또한, 아래의 각 사각형 집합은 인덱스 검색 과정에서 해당 질의 윈도우와 유사하다고 판단하여 검색한 후보 데이터 윈도우들을 나타낸다. 후보 데이터 윈도우를 위한 사각형 내의 각 i-j는 해당 후보 데이터 윈도우가 i 번째 곡의 j 번째 마디에서 시작하는 윈도우임을 의미한다. 여기서 각 윈도우는 네 개의 마디로 구성된다고 가정한다. 모든 질의 윈도우들과 유사한 연속된 후보 데이터 윈도우를 포함하는 것은 7 번째 곡의 5번째 마디로부터 시작되는 멜로디뿐이다(어둡게 표시한 부분 참고). 따라서 이 부분을 제외한 나머지 후보 데이터 윈도우들은 이러한 조건을 만족시키지 못하므로 윈도우 스티칭 단계 후에 모두 제거된다. 이 결과, 후처리에서 실제로 액세스하여 질의 멜로디와의 유사 정도를 계산해야 하는 후보 멜로디들의 수는 크게 줄어든다.

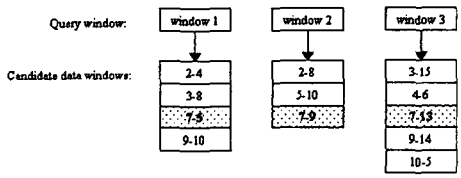

그림 6. 윈도우 스티칭 과정.

4.4. 후처리

후처리는 질의 멜로디와 유사한 멜로디들을 최종적으로 결정하는 과정이다. 이 과정에서는 윈도우 스티칭 단계에서 반환된 후보 멜로디들을 대상으로 디스크에서 액세스하여 질의 멜로디와 실제로 유사한가의 여부를 최종적으로 점검한다. 질의 멜로디와의 실제 유사도가 허용치 ε를 초과하는 멜로디들은 최종 결과에서 제외된다.

5. 성능 평가

본 장에서는 제안하는 기법의 검색 성능을 평가한다. 제 5.1장에서는 실험을 위한 환경을 설명하고, 제 5.2절에서는 실험 결과를 제시한다.

5.1. 실험 환경

본 실험에서는 미디 포맷 1을 따르는 총 500개의 음악 파일들이 저장된 데이터베이스를 사용하였다. 질의 멜로디 생성 방식으로는 데이터베이스로부터 임의의 곡을 하나 선택하여 임의의 위치로부터 시작되는 멜로디를 선택하였으며, 유사 허용치 ε로는 0, 1, 2을 사용하였다. 각 실험 결과로서 길이가 동일한 10개의 질의 멜로디들에 대하여 유사 검색을 수행한 후, 이 응답 시간의 평균을 구하였다. 질의 멜로디의 길이로는 4와 8을 사용하였고, 윈도우 크기로는 2를 사용하였다. 제안된 기법과의 성능 비교 대상으로서 동일한 유사 모델을 채택하되 순차 스캔을 이용하여 유사 멜로디를 검색하는 방식을 선정하였다[4]. 실험을 위한 하드웨어 플랫폼으로는 512M 바이트를 갖는 2GHz 펜티엄 PC를 사용하였다. 또한, 소프트웨어 플랫폼으로는 Linux, mySQL, C++

를 사용하였다.

5.2. 실험 결과

실험 1에서는 4마디로 구성되는 질의 멜로디에 대한 유사 멜로디 검색의 처리 시간을 측정하였다. 그림 7은 실험 결과를 나타낸 것이다. 가로축은 데이터베이스 내에 저장된 곡들의 수를 나타내며, 세로축은 유사 멜로디 검색에 소요된 전체 시간을 나타낸다. IB_SMS(index based similar melody searching)은 제안하는 기법을 이용하여 처리한 시간을 나타내고, Seq는 순차 스캔을 이용하여 처리한 시간을 나타낸다. 또한, 괄호 안의 숫자는 질의에서 사용된 유사 허용치 ε 값을 나타낸다.

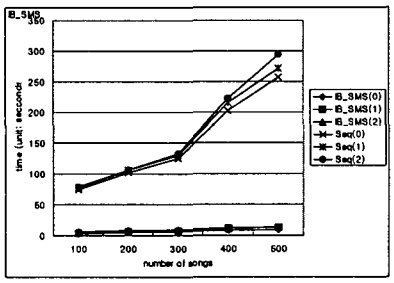

그림 7. 4마디 질의 멜로디에 대한 결과.

실험 결과에 의하면, Seq는 데이터베이스 내의 곡들의 수가 증가함에 따라 처리 시간이 급격하게 증가하는 것으로 나타났다. 이것은 데이터베이스 내의 전체 곡들을 비교 대상으로 하므로 디스크 액세스 시간 및 유사 정도의 비교를 위한 CPU 처리 시간이 증가하는데 따르는 현상이다. 반면, IB_SMS는 데이터베이스 내의 곡들의 수에 큰 영향을 받지 않고, 거의 일정한 처리 시간을 보였다. 이는 제안하는 기법의 인덱스 검색 및 윈도우 스티칭 과정에서 실제로 질의 멜로디와 유사할 가능성이 있는 일부의 멜로디들만을 후보로 걸러주기 때문이다. 제안된 기법의 인덱싱 및 질의 처리 방식으로 인하여 유사 허용치 값에 따라 약 13배에서 28배까지의 성능 개선 효과가 있는 것으로 나타났다.

실험 2에서는 8마디로 구성되는 질의 멜로디에 대한 유사 멜로디 검색의 처리 시간을 측정하였다. 그림 8은 실험 결과를 나타낸 것이다. 전체적인 실험 결과의 경향은 실험 1과 거의 유사한 것으로 나타났으나, IB_SMS와 Seq 모두 4마디로 구성되는 질의 멜로디의 경우와 비교하여 좀더 많은 처리 시간이 소요되는 것을 볼 수 있었다. Seq의 경우, 이것은 멜로디들 간의 유사 정도를 측정하는 시간이 멜로디 길이에 비례하므로 나타나는 현상이다. IB_SMS의 경우, 다음과 같이 해석할 수 있다. 먼저, 질의 윈도우 수가 2배로 되므로 인덱스 검색 및 윈도우 스티칭에서 소요되는 시간이 증가한다. 또한, 후보로 나타나는 멜로디들간의 유사 정도를 측정하는 시간이 멜로디의 길이에 비례한다. 이러한 요인이 전체 소요 시간을 증가시키는 것이다. 순차 스캔을 기반으로 하는 기법과 비교하여 제안된 기법은 유사 허용치 값에 따라 약 18배에서 31배까지의 성능 개선 효과가 있는 것으로 나타났다. 이러한 실험 결과는 제안된 기법의 인덱싱 및 질의 처리 방식이 매우 우수함을 보이는 것이다.

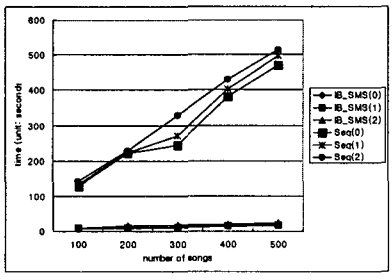

그림 8. 8마디 질의 멜로디에 대한 결과.

4) 기존의 타 기법과의 직접적인 비교는 수행하지 않았다. 그 이유는 제안된 기법은 기존의 기법과 채택하는 유사 모델에 있어 차이가 있기 때문이다.

6. 결론

유사 멜로디 검색은 질의 멜로디와 유사한 멜로디들을 음악 데이터베이스로부터 찾는 연산이다. 본 논문에서는 유사 멜로디 검색을 기반으로 하는 표절 감지 시스템 개발에 관하여 논의하였다. 본 연구에서는 먼저 정합 및 이동 변환을 지원하는 새로운 유사 모델을 제안하였다. 또한, 멜로디의 특징들을 다차원 인덱스의 하나인 R*-트리를 이용하여 인덱싱 하는 방법과 이를 이용하여 인덱스 검색, 윈도우 스티칭, 후처리로 구성되는 삼 단계 질의 처리 방법을 제안하였다. 이러한 연구 결과를 통하여 사용자는 자신이 관심있는 멜로디와 유사한 멜로디를 가지는 곡들을 데이터베이스에서 효과적으로 검색할 수 있다.

제안된 기법의 우수성을 규명하기 위하여 실험을 통한 성능 평가를 수행하였다. 실험 결과에 의하면, 제안된 기법은 순차 검색을 기반으로 하는 단순한 방법과 비교하여 유사 허용치에 따라 약 31배까지의 성능 개선 효과를 가지는 것으로 나타났다. 이는 제안된 기법의 인덱싱 및 질의 처리 전략이 매우 효과적으로 동작함을 보이는 것이다.

7. 감사의 글

본 연구는 2003년도 한양대학교 교내 연구비(HY-2003) 지원과 정보통신부 IT 연구센터(강원대학교 미디어서비스기술 연구센터)의 연구비 지원으로 수행되었습니다.

참고문헌

[1] R. Agrawal et al., "Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases," In *Proc. Int'l. Conf. on Very Large Data Bases*, VLDB, pp. 490-501, 1995.

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

[3] G. H. Cha, C. W. Chung, "A New Indexing Scheme for Content-Based Image Retrieval," *Multimedia Tool and Application*, Vol. 6, No. 3, pp. 263-288, May 1998.

[4] Y. F. Day et al., "Object-Oriented Conceptual Modeling of Video Data," In *Proc. Int'l. Conf. on Data Engineering*, IEEE, pp. 401-408, 1995.

[5] M Flickner et al., "Query by Image and Video Content: The Qbic System," *IEEE Computer*, Vol. 28, No. 9, pp. 23-32, Sept. 1995.

[6] A. Ghias et al., "Query by Humming: Musical Information Retrieval in an Audio Database," *ACM Multimedia*, pp. 231-236, 1995.

[7] R. Hjelsvold and R. Midtsraum, "Modeling and Querying Video Data," In *Proc. Int'l. Conf. on Very Large Data Bases*, VLDB, pp. 686-694, 1994.

[8] J. L. Hsu et al., "Efficient Repeating Pattern Finding in Music Databases," In *Proc. Int'l. Conf. on Information and Knowledge Management*, CIKM, pp. 281-288, 1998.

[9] S. Y. Kim and Y. S. Kim, "An Indexing and Retrieval Mechanism Using Representative Melodies for Music Databases", In *Proc. 2000 Int'l. Conf. on Information Society in the 21st Century*, 2000.

[10] C. C. Liu, P. J. Tsai, "Content-Based Retrieval of MP3 Music Objects," In *Proc. Int'l. Conf. on Information and Knowledge Management*, ACM CIKM, pp. 506-511, 2001.

[11] Standard MIDI File Format, Standard MiDI Files 1.0, Mar. 1988.

[12] E. Oomoto and K. Tanaka, "OVID: Design and Implementation of a Video-Object Database System," *IEEE Trans. on Knowledge and Data Engineering*, pp. 629-643, 1993.

[13] S. W. Smoliar and H. J. Zhang, "Content-Based Video Indexing and Retrieval," *IEEE Multimedia*, pp. 62-71, 1994.

[14] K. Y. Whang, S. W. Kim, and G. Wiederhold, "Dynamic Maintenance of Data Distribution for Selectivity Estimation," *The VLDB Journal*, Vol. 3, No. 1, pp. 29-51, 1994.