MusicXML 전자악보를 위한 K-d 트리 구조 표현

김택헌°, 양성봉
연세대학교 컴퓨터과학과
{kimthun, yang}@cs.yonsei.ac.kr

K-d Tree Structured Representation for MusicXML Music Scores

Taek-Hun Kim°, Sung-Bong Yang
Dept. of Computer Science
{kimthun, yang}@cs.yonsei.ac.kr

요 약

MusicXML은 다양한 전자악보 형식들이 음악을 악보로 표현하는데 있어 지니는 한계를 잘 극복하면서 응용성, 확장성 및 공개성 등의 장점으로 인해 현재 전자악보의 표준으로 가장 적합한 것으로 평가되고 있는 악보 형식이다. 그러나 MusicXML은 XML을 기반으로 한 텍스트 데이터이기 때문에 이러한 악보 형식을 실제 악보로 변환하거나 연주하는 것은 물론 실제 악보 내용을 기반으로 한 악보 검색이 용이하도록 적절한 데이터 구조로 표현하는 것이 필요하다. 본 논문에서는 MusicXML 악보에 대하여 다차원 속성 정보를 가진 데이터의 표현에 용이한 k-d 트리 기반 데이터 구조로 표현하는 방법을 제안한다. 논문은 또한 악보에 대한 k-d 트리 구조를 보다 다양한 응용에 활용할 수 있도록 k-d 트리를 확장하여 구조화하는 방법을 제시한다. 본 논문에서 제안한 방법은 특히 내용을 기반으로 한 악보 정보 검색에 유용하게 이용될 수 있다.

1. 서 론

현재 음악을 기보 형태로 표현하는 다양한 전자악보 형식이 있다. 이러한 전자 악보 형식에는 지금까지 가장 많이 사용되어 온 MIDI 파일 형식을 비롯하여 MusicXML, MML(Music Markup Language), NIFF(Notation Interchange File Format), SMDL(Standard Music Description Language), MNML(Musical Notation Markup Language), MHTML(Music HTML), JScoreML, Xscore(eXtensible Score Language) 등 다양한 음악적 표현과 체계을 갖는 악보 형식이 있다[1][2][3].

그러나 이들 악보 형식은 저마다 표기 형식이 다르고 사용 목적이나 음악적 표현 정도가 각기 다르기 때문에 악보를 다루고자 하는 사람을 중심으로 하지 않고 악보가 처리되는 기계가 중심이 된다거나 악보를 표기하는 기보법이나 분석, 검색 등에 있어 제약을 가지고 있거나, 응용성 및 확장성이 기대에 미치지 못하는 등의 표준화된 전자 악보 형식이 되지 못하는 한계를 가지고 있다[2].

그러나 이들 전자 악보 형식 중에서 지금까지 보편적으로 많이 사용되어져 온 MIDI 악보 형식은 사운드를 직접 다루기 보다는 연주 장치의 정보를 표현하는 방식을 따르고 있어 그 표현 형식이 간결하다. 이는 MIDI 악보 형식 파일의 크기가 작은 장점이 되지만 다른 한 편으로는 사운드의 제어가 명확하게 이루어지지 못하는 단점이 되기도 한다. 또한 MIDI 악보 형식의 경우는 풍부한 음악적 표현을 다양하게 표현하는데 있어서도 한계를 가지고 있다[3].

따라서 이러한 한계를 극복함으로써 다양한 음악적 표현을 가능하게 하면서도 음악에 대한 기보나 분석 및 검색 등 다양한 응용에 적용시키면서 확장성이 높은 전자악보의 필요성이 증대되었고, 인터넷 환경하에서 보다 다양한 이러한 요구를 충족시키기 위하여 MusicXML이라고 하는 XML을 기반으로 하는 전자악보 형식이 개발되었다. MusicXML은 현재 전자악보의 표준으로 가장 적합한 것으로 평가되고 있다.

그러나 음악 악보를 표준화된 전자적인 악보 형식으로 표현하는 일이 중요한 만큼 실 세계에 존재하는 수 많은 악보를 전자 악보로 표현했을 때, 이들 악보들을 악보대로 실제 연주할 수 있도록 하며, 방대한 악보들 중에서 원하는 악보를 검색할 경우에 제목이나 작곡가 등의 단순한 키워드 정보에 의한 것이 아니라 실제 표현된 악보 내용을 기반으로 한 검색을 위한 악보 관리가 무엇보다 중요한 일이 된다.

따라서 본 논문에서는 현재 음악 악보 표현에 있어 가장 표준에 가까운 전자악보 형식인 MusicXML에 대해서 소개하고 이를 검색 및 연주에 용이하게 표현하기 위한 K-d 트리 기반의 데이터 구조를 제안한다. 제안한 방법은 전자악보의 관리 및 악보 연주는 물론 온라인 상에 편재되어 있는 수 많은 악보 데이터를 실제 악보 내용을 기반으로 효율적으로 검색할 수 있는 도구로의 유용성을 지닌다.

본 논문의 구성은 다음과 같다. 2장에서는 MusicXML

악보 형식을 설명하고, 3장에서 k-d 트리구조에 대해서 설명한다. 그리고 4장에서 MusicXML 악보를 표현하기 위한 k-d 트리 구조를 제안하며, 5장에서 결론을 맺는다.

2. MusicXML

미국의 레코다레(Recodare) 사에 의해서 개발된 MusicXML은 연주자들을 포함한 음악인들이 온라인 상에서 악보를 사용할 수 있도록 개발한 XML 기반의 악보 형식으로, 17세기 이후의 일반적인 서양 음악의 기보를 위한 것으로 음악에 대한 기보나 분석, 검색 및 연주를 위한 악보 표기 형식이다[1][2][3][4].

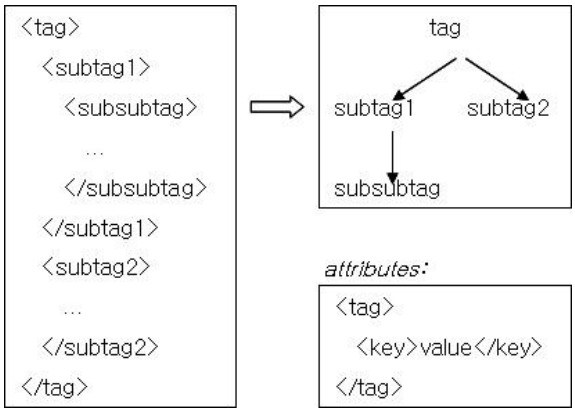


그림 1. XML 트리 구조

```
<score-partwise>
  <identification>...</identification>
  <part-list>...</part-list>
  <part id="p1">
    <measure number="1">
      <attributes>...</attributes>
      <note>...</note>
      <note>...</note>
      <note>...</note>
      <note>...</note>
    </measure>
    <measure number="2">
      <note>...</note>
      <note>...</note>
    </measure>
  </part>
  <part id="p2">
    ...
  </part>
</score-partwise>
```

그림 2. MusicXML 구조

MusicXML은 대표적인 음악 악보 형식으로 지금까지 널리 사용되어 온 MIDI 형식이 풍부한 음악적 표현을 위해서는 제한적이라는 단점을 극복하여 다양한 악보 형식 표현에 용이하고, 또한 DTD가 모두에게 공개되어 있고 그 응용성과 확장성이 뛰어나기 때문에 MIDI의 뒤를 잇는 악보 표준으로 가장 적합한 것으로 평가되고 있다. MusicXML은 현재 버전 1.1이 공개되어 있다.

그림1은 XML의 트리구조를 나타내고 있으며, MusicXML의 구조는 그림 2에서 보여지는 것과 같다[5].

MusicXML은 악보파트방식(partwise)과 악보시간방식(timewise)로 나누어지는데 파트방식을 일반적으로 사용한다. 각각의 파트는 순차적으로 나타내어지며 소절(measure)들로 구성된다. 각 소절은 음표와 박자, 조성, 음자리표 등의 속성 및 나타냄말(direction) 등으로 구성되어진다. 그림 3은 제시된 하나의 음표로 이루어진 악보를 MusicXML로 나타낸 것이다[3][6].

```
<score-partwise version="1.1">
  <part-list>
    <score-part id="p1">
      <part-name>Music</part-name>
    </score-part>
  </part-list>
  <part id="p1">
    <measure number="1">
      <attributes>
        <divisions>1</divisions>
        <key>
          <fifths>0</fifths>
        </key>
        <time>
          <beats>4</beats>
          <beat-type>4</beat-type>
        </time>
        <clef>
          <sign>G</sign>
          <line>2</line>
        </clef>
      </attributes>
      <note>
        <pitch>
          <step>C</step>
          <octave>4</octave>
        </pitch>
        <duration>4</duration>
        <type>whole</type>
      </note>
    </measure>
  </part>
</score-partwise>
```

그림 3. MusicXML 악보 표현 예

3. K-d 트리 구조

K-d 트리는 이진 탐색 트리를 다차원 공간으로 확장한 것으로 여러 개의 속성(attributes)을 동시에 키로 사용하는 색인 구조를 말한다. K-d 트리는 각 레벨에서 모든 속성을 고려하여 트리를 분화하는 것이 아니라 전체 속성의 수를 기준으로 각 레벨에서는 하나의 속성 값을 기준으로 왼쪽 자식 혹은 오른쪽 자식으로의 노드 구성을 결정한다. K-d 트리는 주기억장치에서 동작(In-memory structure)하기 때문에 파일을 이용하는 다른 다차원 색인 구조보다 빠른 탐색을 위한 응용에 이용할 수 있다[7][8][9].
그림 4는 10개의 2차원 점을 a부터 j까지 순서대로 k-d 트리에 삽입한 후에 만들어진 최종 k-d 트리의 예를 보여준다[7].

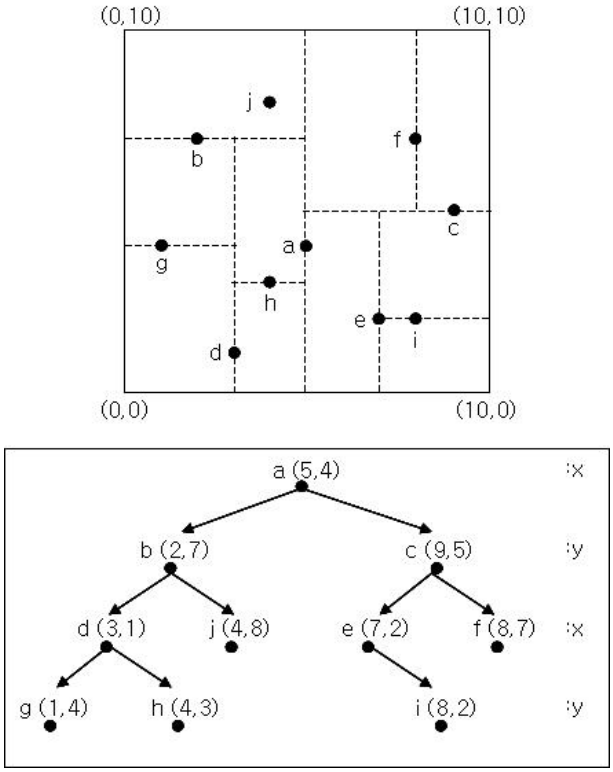


그림 4. K-d 트리의 생성

K-d 트리에서 임의의 한 노드를 검색해 나가는 것은 이진 탐색 트리에서와 유사하지만 k-d 트리의 경우는 각 레벨에서 비교하는 속성이 항상 같지 않다. 즉, 속성이 하나인 경우는 이진 탐색 트리와 같지만, 속성이 여러 개인 경우는 트리의 루트에서 하나의 속성을 비교하였다면 그 다음 레벨에 와서는 다른 속성에 대한 비교를 수행한다. 이렇게 하여 레벨을 달리 할 때마다 서로 다른 속성을 이용하여 원하는 노드를 검색해 나간다. 이 때, 모든 속성을 모두 이용하였으나 원하는 노드를 아직 찾지 못하였을 때에는 다시 처음 비교했던 속성부터 차례로 비교해 나간다. 이렇게 함으로써, 다중 속성 데이터에 대한 트리 구성과 탐색 연산이 가능하게 된다.

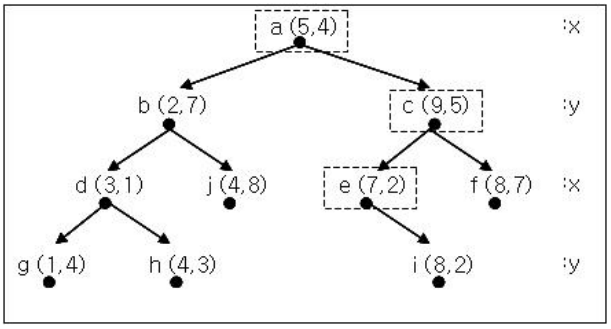


그림 5. K-d 트리의 탐색

그림 5는 그림 4에서 생성된 k-d 트리에서 점 e(7,2)를 탐색해 가는 과정을 보여준다. 탐색 과정은 먼저 탐색할 점 e(7,2)을 루트인 a(5,4)와 x 좌표 속성에 대해 비교한다. 이 때, 7>5 이므로 탐색은 오른쪽 서브 트리인 c(9,5)로 이동한다. 점 c는 루트 다음 레벨에 있는 점이기 때문에 찾을 점 e와 점 c의 비교는 점의 다른 속성, 즉, y 좌표 속성에 대해 비교하여야 한다. 따라서 두 점의 y 좌표를 비교하면, 2<5 이므로 탐색은 다시 왼쪽 서브 트리로 이동하게 된다. 이렇게 이동한 새로운 레벨에서 점 e(7,2)와의 비교는 다시 x 좌표 속성에 대해서 비교하게 된다. 이것은 주어진 점의 속성이 x와 y, 두 가지로만 이루어져 있기 때문에 모든 속성에 대한 비교 후에는 다시 처음 속성부터 차례로 비교해 나가게 된다. 탐색은 점 e(7,2)에 이르렀을 때 원하는 점을 찾게 됨으로 검색을 마치게 되며, 이 때, 검색은 성공하게 된다.

4. MusicXML 악보의 K-d 트리 구조 표현

본 논문에서 제안하는 MusicXML 악보의 k-d 트리 구조 표현 방법은 다음과 같다. 먼저 MusicXML 악보에 대한 처리 흐름을 살펴보고, 악보에 대한 일반적인 k-d 트리 구성법을 제시한다. 그리고 이러한 트리 구성에서 나타나는 문제점들을 해결하기 위한 방법으로 k-d 트리 구성을 확장하여 나타낸다.

4.1. 악보 처리

MusicXML 전자악보는 그 자체로는 여러가지 종류의 태그(tag) 정보들로 이루어진 하나의 XML 텍스트 데이터이다. 따라서 이것을 프로그램에 의해 연주할 수 있도록 하거나, 연주자 등 실제 사용자들이 알 수 있도록 오선지에 그려진 악보로 제공하기 위해서는 XML 악보에 대한 분석을 통한 변환 과정이 필요하다. 이 때 분석을 위해서는 MusicXML 악보를 파싱(parsing)하여 각각의 태그 정보가 실제 악보에서

어떤 기능을 하는지 여부를 가려내야 한다.

MusicXML 악보는 여러가지 구성요소들로 이루어져 있으며 이들 구성요소들은 머리(header)와 몸체(body)의 두 부분으로 크게 나누어진다. 이 때, 머리 부분에는 작품 제목, 작품 번호, 작곡자, 작사자, 파트 목록 등 악보에 대한 상위 정보들이 나타나게 되며, 몸체 부분에는 악보의 실질적인 내용이라 할 수 있는 조표, 박자, 빠르기표, 음표, 쉼표 등 악보상의 음악을 표현할 수 있는 다양한 구성 요소들에 대한 세부 정보들로 이루어진다.

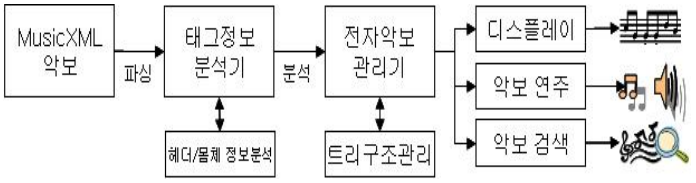


그림 6. 악보 처리 흐름도

그림 6은 MusicXML 악보의 처리 흐름도를 보여주고 있다. XML 악보는 파싱을 통해 태그 분석 정보를 분석한다. 이 때, 태그 분석은 악보의 헤더 정보 및 몸체 정보에 대한 사항을 포함한다. 태그 분석기에 의해 처리된 분석 정보는 전자악보 관리기로 넘어오게 되며, 이 관리기에서 변환된 악보 정보에 대한 k-d 트리 구조를 구성하게 된다. 악보 관리기는 악보에 대한 트리 데이터 구조를 관리하며 악보를 실제 화면에 보여주는 디스플레이 모듈, 악보를 컴퓨터의 음원을 이용하여 실제 연주해 주는 연주 모듈 및 원하는 악보를 실제 멜로디 정보를 이용하여 찾을 수 있는 내용 기반 검색을 할 수 있도록 한다.

4.2. K-d 트리 구조 표현

태그 정보 분석기에 의해 분석된 악보가 그림 7과 같을 경우에 악보 관리기는 이 악보에 대한 k-d 트리를 구성하게 된다. 여기서 악보의 속성 정보로는 악보를 이루는 가장 중요한 요소라고 할 수 있는 음의 높이와 길이를 이용한다. 이 때, 음의 길이는 다시 음표와 같은 길이의 쉼표 정보를 이용하며, 음표와 쉼표는 동일한 속성으로 간주한다. 그림 8은 음의 높이 속성을, 표 1은 음의 길이 속성에 대한 각 속성 값을 나타내고 있다. 여기서 음표 및 쉼표는 표1에 주어진 것 외에 여기에 점이 포함된 점음표와 점쉼표를 포함한다. 점음표와 점쉼표는 해당 음표와 쉼표의 1.5배의 길이를 갖는다.

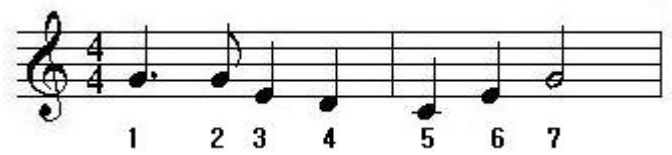


그림 7. 악보 예제

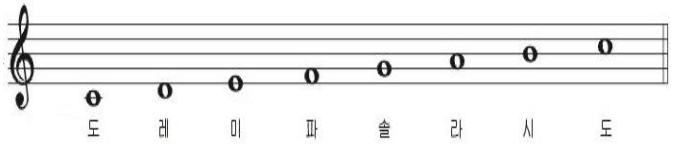


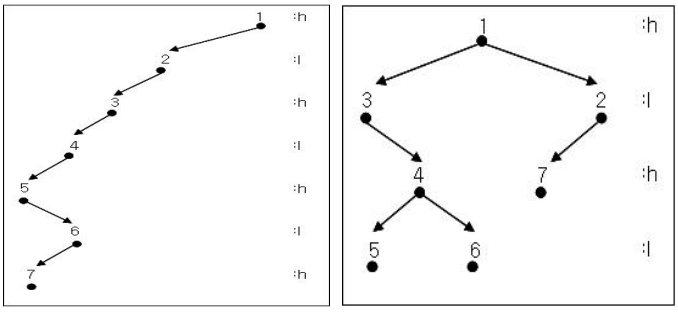
그림 8. 음의 높이 : 기본음계

음표	이름	박자	쉼표	이름
	2분음표	2박		2분쉼표
	4분음표	1박		4분쉼표
	8분음표	1/2박		8분쉼표
	16분음표	1/4박		16분쉼표

표 1. 음의 길이 : 음표와 쉼표

아래 그림 9는 그림 7에 나타난 악보를 k-d 트리로 구성한 결과를 나타낸 것이다. K-d 트리의 구성은 루트에서 음의 높이 속성을 비교하고 다음 레벨에서 길이 속성을 비교하며, 이 두 가지 속성을 모두 비교한 후에는 다시 같은 속성에 대하여 비교를 반복한다. 그림에서 h는 높이 속성을 l은 길이 속성을 나타낸다. 트리를 구성할 때 각 레벨에서 비교할 값이 서로 같을 경우가 생기게 되는데, 이 때, 같은 속성 값에 대해서 왼쪽 자식으로 분화하도록 한 것을 그림 9-(a)에 나타내었고, 그 반대의 경우를 그림 9-(b)에 나타내었다.

그림에서 보여지는 것과 같이 서브 트리 생성 규칙에 따라 같은 악보라 하더라도 구성되는 트리의 모양이 전혀 다르게 된다. 또한 경우에 따라서 (a)에서와 비슷하게 한쪽으로만 기울어진 트리가 생성되기도 하며, 이런 경우 트리의 깊이는 최대 음의 개수만큼 늘어나게 된다.



(a) (b)

그림 9. 악보(그림 7)에 대한 k-d 트리 구성

4.3. K-d 트리 구조 표현의 확장

그림 9에서 생성한 k-d 트리는 주어진 악보에 대해서 차례대로 트리를 구성하였으나, 트리를 구성한 이후에

보여지는 결과 트리로부터는 완전 편향 트리로 구성된 경우를 제외하고는 악보의 순서를 알 수가 없게 된다. 따라서 이렇게 구성된 악보 트리는 하나의 음표에 대한 트리 탐색 이외에 악보에 따른 순서적 연주나 악보에 대한 부분적인 내용 검색 등은 할 수 없게 된다. 또한 완전 편향 트리로 구성된 경우도 단지 순차적 표현 구조를 가지게 되어 악보에 대한 내용기반 검색을 할 수 없게 된다. 따라서 이러한 문제들을 해결할 수 있는 다른 방식의 트리 구성이 요구된다.

그림 10은 이러한 문제들을 해결하기 위해 k-d 트리를 확장하여 구성한 것이다. 악보를 k-d 트리로 구성하기 위해 사용하는 악보 속성은 4.2절에서와 같다. 또한 트리의 루트에서부터 차례로 비교해 나가는 속성도 같은 순서를 유지한다. K-d 트리를 구성하는 방법은 악보의 모든 음표에 대하여 동일하다.

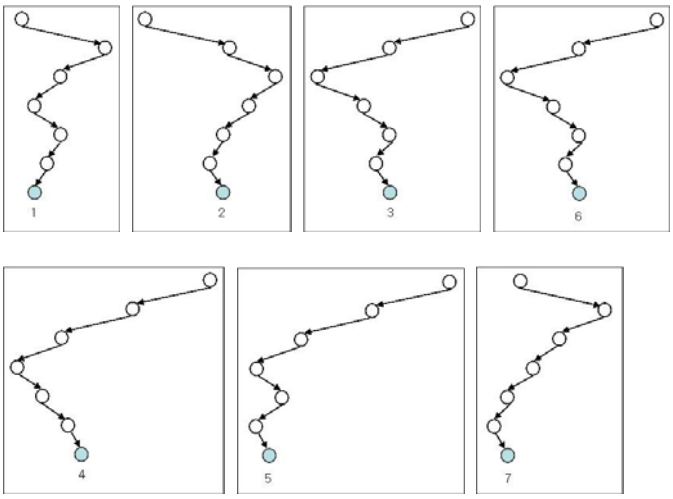


그림 10. 악보(그림 7)에 대한 확장된 k-d 트리 구성

그림 10의 첫 번째 트리는 그림 7에 나타난 악보의 첫 번째 음표, 즉, 1번 음표에 대한 k-d 트리 구성을 나타낸다. 그림에서 1번 음표에 대한 각 속성에 대한 속성 값은 음의 높이 속성에 대해서는 '솔' 값을 가지며, 음의 길이 속성에 대해서는 '점4분음표' 값을 갖는다. 이러한 속성 값을 갖는 1번 음표에 대한 k-d 트리 구성시 속성 값의 비교 대상은 이제는 다른 음표와의 비교가 아니라 속성 값 전체를 대상으로 한다. 즉, 전체 속성 값 중에서 주어진 속성 값을 이진 탐색 과정을 거쳐 찾아가면서 값을 비교해 나가게 된다. 이때, 각 수준을 따라 내려가면서 비교하는 속성은 서로 다른 속성에 대해 모두 비교한 이후 다시 반복 수행한다.

그림 11은 음의 높이 속성에 대한 1번 음표의 비교 순서를 나타낸 것이다. 먼저 전체 속성 값들 중에서 1번 음높이 속성 값인 '솔'은 오른 쪽 절반에 해당한다. 따라서 이 경우에 k-d 트리의 생성은 오른쪽 서브 트리로 내려오게 된다(그림 10의 첫 번째 그림에서 루트 노드에서 오른 쪽 자식 노드로 내려오는 것). 다음 수준에서는 음의 길이에 대해 비교하여 왼쪽 자식 노드로 이동하였고, 다시 음의 높이 속성에 대해서 비교한다. 이 때에는 먼저 계산하고 남은 오른 쪽 절반 속성 값들 중에서 왼쪽 절반에 해당 속성 값이 있기 때문에 왼쪽 자식 노드로 향하게 된다. 이렇게 하여 서로 다른 속성들에 대해서 레벨을 따라 내려오면서 번갈아 가면서 비교 연산을 수행하고, 해당 속성에 대해서는 전체 속성 값들에 대해 차례대로 비교해 가면서 해당되는 속성 값을 찾게 되면 최종적 위치에 주어진 음표 정보를 삽입할 수 있게 된다.

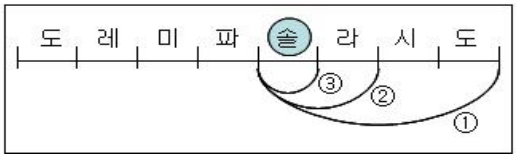


그림 11. 음의 높이 속성 비교 순서

그림 10의 나머지 트리들은 그림 7의 악보에서 보여지는 각기 다른 번호들에 대한 k-d 트리를 동일한 방법으로 각각 구성한 것이다. 이렇게 각각의 트리를 구성한 후에는 생성된 모든 트리의 루트 노드를 통합하여 자식 노드들을 그 방향을 중심으로 통합된 k-d 트리로 구성한다. 그림 12는 이렇게 통합된 하나의 k-d 트리를 보여주고 있다.

구성된 트리를 보면 실제 정보는 트리의 마지막 리프 노드에 들어 있는 것을 알 수 있으며, 내부 노드들은 트리를 탐색하기 위한 인덱스의 역할을 하게 된다. 이 때 k-d 트리에서 실제 악보 정보들은 동일한 수준의 리프 노드에 존재하는 것이 보장된다. 이 때, 리프 노드에 삽입된 정보는 악보에 따른 순서대로 링크가 구성되어져 순차 처리가 용이하게 된다. 또한 트리의 속성이 (a_1, a_2, …, a_n)이고, 각 속성에 대한 속성 값이 각각 (2^{v1}, 2^{v2}, …, 2^{vn})이라고 할 경우, 트리의 깊이는 최대 $\sum_{i=1}^{n} v_i$ 만큼임을 보장한다.

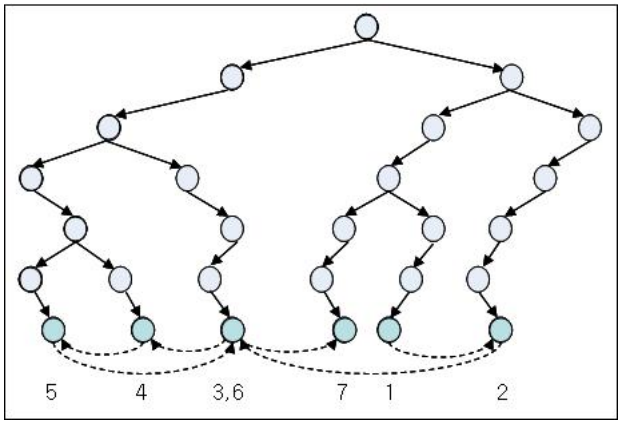


그림 12. 악보의 k-d 트리 구조 표현

5. 결론 및 향후계획

음악을 기보 형태로 표현하는 다양한 전자악보 형식들 중에서 지금까지 MIDI가 가장 대표적인 악보 형식으로 사용되어 왔다. 하지만 다양한 음악을 악보로 표현하는데 있어 지니는 여러 가지 한계를 극복할 필요성이 점차 증가되었으며, 이러한 요구에 가장 적합한 악보 형식으로 MusicXML이 개발되었다. MusicXML은 인터넷 환경하에서 보다 다양한 요구를 충족시키기 위해 개발된 XML을 기반으로 한 악보 형식으로, 응용성, 확장성 및 공개성 등의 장점으로 인해 현재 전자악보의 표준 형식으로 가장 적합한 것으로 평가되고 있다.

그러나 XML이라는 텍스트를 기반으로 한 MusicXML 악보 형식에 대해서 실제 악보로 변환하여 표현하고 연주하는 것은 물론 다양한 악보 검색에 이용하기 위해서는 악보 형식을 적절히 데이터 구조화하고 이를 관리하는 것이 필요하게 된다.

본 논문에서 우리는 k-d 트리 데이터 구조를 이용하여 MusicXML 전자 악보를 표현하는 방법을 제안하였다. K-d 트리 구조는 다차원 속성 정보를 가진 데이터를 표현하는데 용이한 데이터 구조로써, 다양한 속성 정보를 포함하는 악보 표현에 적절하다. 우리는 또한 악보에 대한 k-d 트리 구조를 다양한 응용에 이용할 수 있도록 k-d 트리를 확장하여 표현하는 방법을 제시하였다. 이렇게 확장된 k-d 트리는 악보 검색에 특히 유용하게 이용할 수 있다.

향후 계획으로는 제안한 k-d 트리 데이터 구조를 확장하여 추상화하고 필요한 연산 기능을 정립하고자 한다. 또한 본 논문에서 제시한 악보 처리 흐름도에 따른 각 모듈에 대한 기능을 세분화하여 하나의 통합된 악보 관리 및 검색 시스템을 제안하고자 한다.

참고문헌

[1] 김병선, 민요 데이터에 대한 MusicXML의 적용 가능성 및 활용 방안, 한국 민요학회 제11차 전국학술대회, 2004.

[2] 김정희, 곽호영, 통합 방식을 이용한 대보표 악보의 XML 스키마 모델, 한국해양정보통신학회, Vol. 7, No. 2, pp. 302-313, 2003.04.

[3] Michael Good, MusicXML: An Internet-Friendly Format for Sheet Music, The XML 2001 Conference Proceedings, 2001.12.

[4] Recordare: Internet Music Publishing and Software: http://www.recordare.com/

[5] Craig Stuart Sapp, MusicXML, Lecture Notes of Music253/CS275A, Stanford University, 2005.

[6] Michael Good, MusicXML in Practice: Issues in Translation and Analysis, The MAX 2002 conference on Musical Applications Using XML, 2002.9.

[7] 문양세, Multidimensional Index Structures, Lecture Notes of Advanced File Processing, 강원대학교 컴퓨터과학과, 2006.

[8] Jon Louis Bentley, Multidimensional Binary Search Trees Used for Associative Searching, Communications of the ACM, Vol. 18, No. 9, 1975.9.

[9] Hanan Samet, Foundations of Multidimensional and Metric Data Structures, Morgan Kaufmann, 2006.

[10] David Rizo Valero, Jose Manuel Inesta, Tree-structured Representation of Melodies for Comparison and Retrieval, Int. Workshop on Pattern Recognition in the Information Society, PRIS 2002, pp.140-155, 2002.

[11] David Rizo, Jose Manuel Inesta, Francisco Moreno-Seco, Tree-structured representation of musicial information, 1st Iberian Conference on Pattern Recognition and Image Analysis, pp. 838-846, 2003.

[12] Jeremy Kubica, Joseph Masiero, Andrew Moore, Robert Jedicke, Andrew Connolly, Variable KD-Tree Algorithms for Spatial Pattern Search and Discovery, Neural Information Processing Systems, 2005.12.