A Feature Extraction and Classification Method for Music Objects in MusicXML[1]

Yu-Chih Shen(沈裕智) and **Jia-Lien Hau(徐嘉連)[2]**
Department of Computer Science and Information Engineering,
Fu Jen Catholic University, Taiwan, R.O.C.

alien@csie.fju.edu.tw

摘要

在音樂資訊查詢（music information retrieval）領域中，「從音樂物件中擷取出代表內涵（content）的音樂資訊作為特徵值是重要的研究重心」。從音樂物件中擷取出特徵值之後，可以作為更有效的音樂分析與查詢，並提供更多樣化的服務。

在這篇論文裡面，我們的資料格式是MusicXML。根據樂理（music theory）與曲式（music form）的理論為基礎，從古典樂曲的音樂物件中，擷取出可以呈現音樂結構的特徵。參考「曲式」的結構特徵（structural feature）之階層性規則（hierarchical rule）特色，將這擷取出來的音樂特徵，建立出一棵樹狀資料結構，構成一棵具有樂理特徵、樂理結構的曲式樹（MF-tree）。同時，我們也提出曲式樹的不相似函數（dissimilarity function），利用曲式樹計算出音樂物件的不相似度。最後，採用階層式分群（hierarchical clustering）與系統實作來展示分群的效果。

關鍵詞：特徵擷取（feature extraction），曲式樹（MF-tree），階層式分群（hierarchical clustering）。

1 . 序論

音樂資訊查詢系統（music information retrieval），是近幾年來快速發展的一個新穎研究領域。藉著音樂特徵的擷取與應用，系統可以提供更多的服務給使用者，如：推薦（recommendation）、哼唱查詢（querying by humming）。因此，特徵的擷取就成為重要的研究重心。

1.1 動機

在音樂資訊查詢領域中，「從音樂物件中擷取出代表內涵（content）的音樂資訊作為特徵值是重要的研究重心」。在早期的論文研究中，對於音樂資訊查詢系統最常使用的方法是：將一首音樂的旋律（melody）轉換成字串（string）表示，經由字串比對的方式，得到音樂的相似度。一般而言，主要的資料來源有兩種：一種是 symbolic 格式，如：midi。一種是 wav 格式，如：mp3、CD audio。但是，當初 midi 的設計是為了連接多樣化的裝置（connecting various device），只儲存基本的音樂字串，如：旋律字串（melody string）、節奏字串（rhythm string），卻沒有儲存關於音樂結構的相關訊息。因此，我們資料來源改採用目前普遍的MusicXML，作為我們音樂資料的來源。

關於音樂的結構特徵（structural feature），最顯著的特色有：重覆性規則（repeating rule）與階層性規則（hierarchical rule）。重覆性規則與階層性規則皆可以呈現出音樂曲式結構的特性。[Hsu01]利用重覆性規則找尋音樂物件中的重覆樣型。然而，目前就我們所知，尚未有利用階層性規則來表示音樂結構的研究。所以，我們提出一個想法，不只是單純地利用字串表示音樂特徵，而是加上結構的概念。參考曲式的階層性規則，擷取出新的音樂特徵。將古典樂曲的音樂物件，建構成一棵具有樂理特徵、樂理結構的樹狀結構，我們稱之為曲式樹（MF-tree）。並且，可以作為音樂資訊系統的有效分類或是其他相關應用。

接著，關於音樂系統的相關應用，舉例來說：（1）以特徵擷取作為分類（feature extraction for classification）：藉由樂理特徵的擷取可以作為音樂分類的應用。（2）找尋重覆的結構樣型：

[1]本論文研究為國科會補助之研究成果，計畫編號為 NSC-93-2213-E-030-005。
[2] 本論文的相關聯絡人。(To whom all correspondence should be sent.)

藉由重覆結構樣型（repeating structure pattern）內涵的音樂資訊，得到重覆樣型音樂字串片段。（3）音樂查詢（music search）：幫助音樂查詢系統的設計，加快查詢速度。（4）音樂推薦（music recommendation）：促進音樂系統的推薦。（5）音樂摘要（music summary）：藉著音樂結構的呈現，找尋重要的結構片段，進而作為音樂摘要的呈現。（6）音樂分析（music analysis）：針對特定的音樂作曲家，執行音樂作品的分析。（7）拷貝偵測（copy detection）：藉由音樂結構的比對與相似計算，搜尋出結構相似與旋律相似的拷貝作品。

1.2 音樂特徵的概述

針對音樂物件的特性，目前音樂特徵的擷取，從 Hsu [Hsu01]我們可以得知，音樂的特徵值可以分成以下四類：

（1）「Static music information 」是指音樂物件中，所包含的基本特性（keyword-based description），包括：調性、節拍、演奏速度等。例如，「Beethoven's Symphony No. 5, Op. 67」的 static music information 是 C minor, 4/4 等。

（2）「Acoustical features」包括 *loudness*、*pitch*、*duration*、*bandwidth* 和 *brightness* 等聲音特徵值。從音樂物件中的原始資料（raw data）中，經由簡單的計算可以得到這些特徵值。例如，音樂物件的 loudness 是計算音響訊號（audio signal）的方均根值（root-mean-square value ）所得，常用的單位是「分貝（decibel）」[Wold96]。

（3）「Thematic features」則是「旋律（*melody*）」、「節奏（*rhythm*）」、「和弦（*chord*）」等，從類似樂譜（score-like）的資料格式中求得的特徵值。並且此類的特徵值經常以字串表示。例如，"G-G-G-E"是一音樂物件的旋律字串、"C-Am-Dm-G7" 則是和弦字串。

（4）「Structural feature」：從「音樂曲式學」的研究成果得知，古典音樂通常在作曲時會依照一些規則而構成「曲式（music form）」。其中，最顯著的特色是：「階層性規則（hierarchical rule）」和「重覆性規則（repetition rule）」[Jone74][Krum90] [Narm90]。階層性規則是指在古典音樂中，會根據特定形式、由上往下所組成，舉例來說：一個音樂物件包含了樂章，樂章包含樂段，樂段包含樂句，樂句包含音形（a music object consists of *movements*, a movement consists of *sentences*, a sentence consists of *phrases*, and a phrase consists of *figures*）[Hsu01]，如圖 1 所示。重覆性規則是指在古典音樂中，通常會有一個或數個「動機（*motive*）」，不斷地重覆出現在整個音樂中。除了古典音樂之外，在其他類型的音樂中，也可以觀察到重覆性規則。舉例來說：在流行音樂中，經常出現的「副歌（*refrain*）」，就是一個符合重覆性規則的例子。

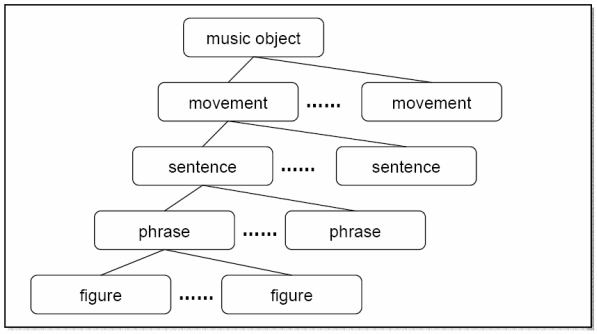

圖1. 音樂物件的階層性規則（The hierarchical rule of a music object）

1.3 相關研究

目前的相關研究，我們分別針對音樂特徵的擷取，音樂的辨識（music recognition），音樂的分類以及 XML 的結構相似（structure similarity）來作介紹：

關於音樂特徵擷取的研究，Miura[Miur03]介紹旋律的音高頻譜（pitch spectrum），利用音高頻譜來評量相似度，並且，可以容易地判斷出歌曲中的移調（transposition）。Wai[Wai03]對於多音音樂資料庫（polyphonic music database），提出一個串流分割演算法（stream segregation algorithm），將數個音高分攤成數個串流（stream），呈現出旋律線（melody line）概念的音樂特徵。Hsu[Hsu01] 定義出「不明顯的重覆樣型（nontrivial repeating pattern）」的音樂特徵，利用關係矩陣（correlative matrix） 和 RP-tree 快速地找到最長的重覆樣型（exact repeating patterns），Hsu[Hsu04]更進一步提出找尋近似的重覆片段，經由切割（cut）與樣型合併（*pattern_join*），設計出一個「類似層次（level-wise）」的方法來找尋近似的重覆片段。

對於音樂曲式結構的辨識，Liu[Liu02]以 Johann Strauss waltz centos 為範例，根據樂理提出了三種假設。以四個小節為單位，設計出一個新的結構（稱之為：Melody-tree）。一棵 Melody-tree 是一棵三層的樹，利用 Melody-tree 作為曲式識別（form recognition）的重要特徵。Seifert[Seif03]則利用語意的關係（semantic relationship），根據專家決定出重要的音樂片段（稱之：characteristic motifs），設計出 Leadsheet-tree，作為音樂的辨識。

音樂分類的研究中，Shan[Shan02]利用和弦（chord）特徵研究不同的特徵表示法（feature representations），設計出一個音樂風格（music style）的發掘和分類方法，幫助音樂查詢系統發掘音樂風格。Kuo[Kuo02] 提出一個音樂過濾系統（music filtering system），根據使用者的行為（behavior），利用多重型態旋律風格（multi-type melody style）的分類方法發掘出旋律樣型（melody patterns），推薦音樂物件給使用者。Kuo[Kuo04]針對旋律風格的查詢，提出四種查詢的類型，查詢結

果會提供給使用者新穎的（new）或是不知道（not known）的樂曲。

關於 XML 結構相似的研究，Wang[Wang04]利用結構資訊（structural information），提出一個階層式演算法（hierarchical algorithm，命名為：S-GRACE），減少當 XML documents 儲存在關連式表格（relational tables）查詢時所需要的合併成本（join cost）。Canfora[Canf04]提出一個 XML 相似計算的方法，作為資訊檢索系統（Information Extraction Systems，*IES*）的評估。Bertino[Bert04]對於 XML document 與 DTD 利用結構相似度（structural similarity），提出一個比對演算法（matching algorithm），以及相關的應用，如：XML documents 的分類、DTD structure 的評估、結構性的查詢、XML documents 的選擇性傳播（selective dissemination）、XML document 內容的保護（protection）。Lee[Lee02]進一步地利用結構相似度結合語意（semantics）的概念，提出 XClust algorithm，有效地整合 DTD schema。

1.4 問題的描述

我們的研究目的是針對 MusicXML 格式，從古典樂曲的音樂物件中，擷取出音樂特徵（feature extraction），並藉此進一步地做分類應用（classification）。我們面臨的第一個問題是：從古典樂曲的音樂物件中，擷取出音樂結構特徵，來呈現音樂的結構，並建構出一棵曲式樹。第二個問題是：曲式樹的不相似度計算法，以及分類的方法。

這篇論文的內容組織如下：第二章為介紹我們所提出的方法，其中包括 MusicXML document 的介紹、系統流程的說明、曲式樹建構的方法、曲式樹的不相似度計算法，以及階層式分群的方法。第三章是我們的系統架構與系統介面。而第四章是實驗結果。最後，第五章則是本篇論文的結論與未來工作。

2. 方法的介紹

在這一個章節，我們首先會介紹 MusicXML document，然後是整個方法的系統流程。接下來，提出建構曲式樹的方法，曲式樹的不相似度計算法。最後是階層式分群的方法。

2.1 MusicXML 的介紹

XML 是在資訊交換上，目前廣泛被使用的一種資料格式。在這樣的趨勢下，以 XML 的技術為基礎，發展出一種新的音樂資料格式稱之為 MusicXML。MusicXML 的設計目的是：為了音樂資訊的分析與擷取。MusicXML 是一種儲存樂譜（score-like）的資料格式，包含了符號式（symbolic）的音樂資料，以及許多描述資料的資料（metadata）。如：作者（creator）、譜號（clef）、調性（key）、節拍（beat）、小節（measure）、音高（pitch）、音長（duration）、旋律（melody）、節奏（rhythm）、和弦（chord）、圓滑線（slur）、音符類型（type）、裝飾記號（grace）、歌詞（lyrics）…等的音樂相關資訊。

首先，我們先舉個簡單的例子來說明一個 MusicXML document 的結構。從布拉姆斯（Johannes Brahms）的「旋律（Wie Melodien Op.105）」中，選取出的一段樂譜（如圖 2 所示）。這段樂譜的第一部（voice）之第一小節在 MusicXML document 中（如圖 3 所示），相當於節點<part id="P1">包含的子節點之第一小節。在圖 2 中，第一小節的第二個音符在 document 的格式，相當於在圖 3 中"矩形"框住的部份，節點<measure number="1">的第二個子節點<note>，所形成的樹狀結構，就是在圖 4 中"矩形"框住的部份。

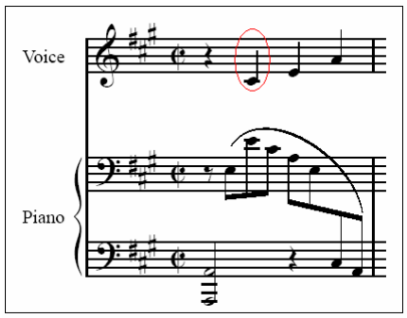

圖2. 布拉姆斯的「旋律（Wie Melodien Op.105）」之第一小節

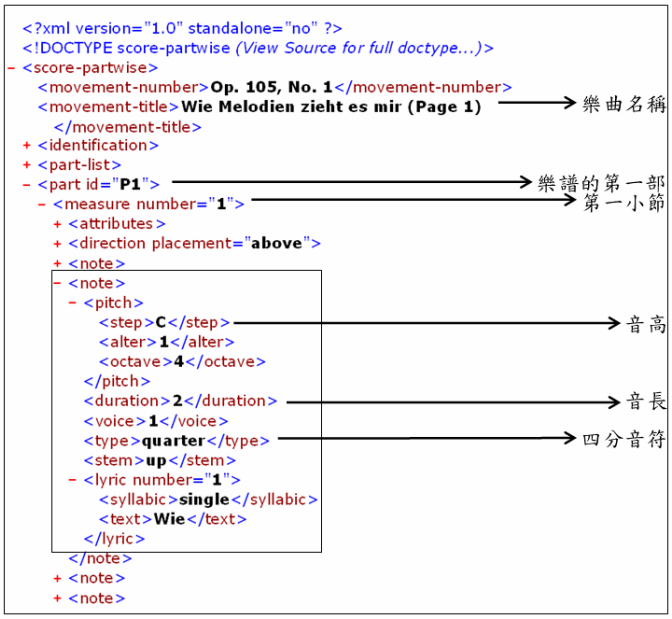

圖3. A MusicXML document

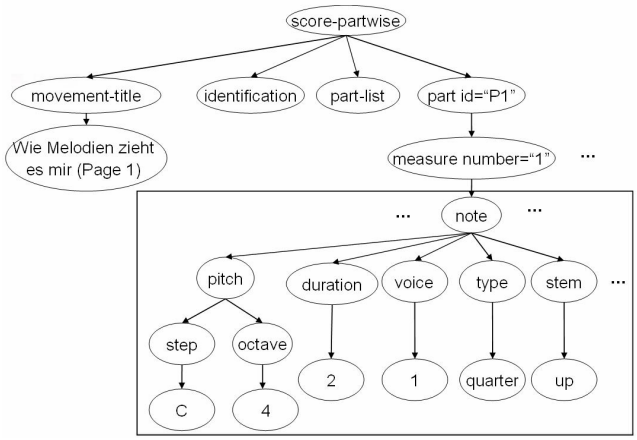

圖4. The structure of MusicXML document

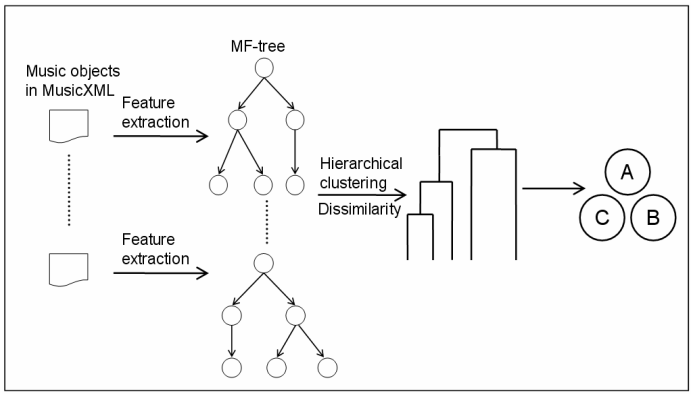

圖5. 系統流程

接下來，對於 MusicXML document 有了基本的瞭解之後，我們先說明不採用目前計算 XML document 相似度方法的原因。由於，我們的資料格式是 MusicXML document，相對於目前的 XML document 的差別，在於結構（structure）的不同。我們可以從網際網路中，獲得許多的 XML document。從這些 XML document 中，我們可以觀察到一件事：在眾多的 XML document 中，每一個 XML document 的結構都是不相同的，各自有各自獨特的結構。然而，目前的許多研究方法，都是以結構來計算 XML document 的相似度[Wang04] [Canf04] [Bert04] [Lee02]。但是，對於 MusicXML documents 來說，大部分的 MusicXML document 都有相似的階層與高度。如圖 4 所示，score-partwise 會包含 movement-title、identification、part-list、part id="P1"，part id="P1"包含 measure number="1"，measure number="1"包含 note，note 包含 pitch、duration、voice、type、stem…等相似的結構。在這樣的情況下，如果，採用目前已有的 XML document 相似度計算方法，我們無法從結構中，求得音樂物件的相似度。所以，我們也將在本篇論文中的第 2.4 節，提出一個適合樹狀結構的不相似度計算法，得到音樂物件的不相似度。

2.2 系統流程

系統流程的介紹，如圖 5 所示。首先，我們會針對 MusicXML 格式的古典音樂物件，參考曲式的階層性規則，得到音樂特徵，並建構出一棵曲式樹。接著，提出曲式樹的不相似度計算法。最後，由階層式分群得到分群的結果。

2.3 曲式樹（MF-tree）的建構

我們知道 MusicXML 是一種描述音樂資訊的格式，但是 MusicXML 包含了許多與樂曲曲式結構不相關的音樂資訊，如：作者（creator）、譜號（clef）、調性（key）。所以，我們參考曲式的階層性規則，針對 MusicXML 格式，古典樂曲的音樂物件中，建構出一棵具有樂理特徵、樂理結構的曲式樹（MF-tree）。一棵曲式樹包含的音樂特徵，可以呈現出音樂的結構，又不會失去音樂本身的內涵，如：旋律、節奏。

一個音樂物件會建構出一棵曲式樹。對於一個音樂物件而言，一個音樂物件就是一首古典樂曲。一首古典樂曲，可以是由一個樂章或是多個樂章所組成。舉例來說：一首古典交響樂曲，可以分成多個樂章，如：貝多芬的「C 小調第五號交響曲（命運交響曲）」，共分成四個樂章。另外，如果一首古典樂曲沒有所謂的樂章，則我們將這一首古典樂曲視為一個樂章，如：布拉姆斯的「旋律（Melodien Op.105）」就是一首的古典樂曲，並沒有分所謂的樂章部分，所以，我們將這首古典樂曲視為一個樂章。

因此，我們將一個古典樂曲的音樂物件，並參考曲式的階層性規則，擷取出「音樂物件（music object）」、「樂章（movement）」、「片斷（segment）」、「子片斷（subsegment）」等特徵值，建構出一棵具有樂理特徵、樂理結構的樹（命名為：MF-tree），如圖 6 所示。一棵曲式樹包含音樂物件、樂章、片斷、子片斷…等節點，好比是一篇文章（document）包含「段落（paragraph）」、「句子（sentence）」、「單字（word）」。曲式樹的結構概念就類似於文章的結構概念。

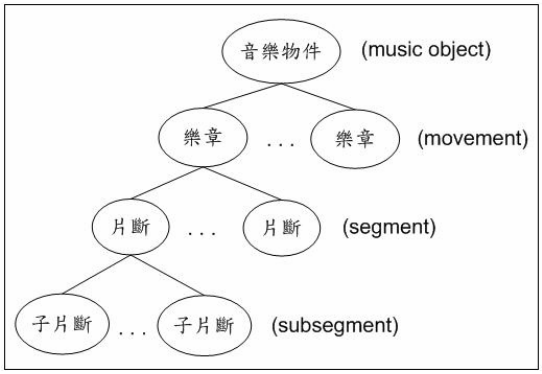

圖6. 曲式樹的結構

另外，對於 MusicXML 來說，如：貝多芬的 C 小調第五號交響。則利用四個 MusicXML document 來儲存，一個 MusicXML document 表示一個樂章。然而，就布拉姆斯的旋律（Wie Melodien）來說，這一個音樂物件就是一個樂章，利用一個 MusicXML document 來儲存。一個音樂物件就是一個樂章由一個 MusicXML document 來儲存。所以，在我們的實作方面（本篇論文的第 3 章節），目前是以一個 MusicXML document 為主，一個音樂物件，就是一個樂章，由一個 MusicXML document 來儲存。

接下來，我們先介紹曲式樹（MF-tree）的定義，然後，介紹建構曲式樹的方法。

Definition 1 ：曲式樹是一棵四層的樹，並擁有以下三個性質：

Property 1. 一棵曲式樹有四層。從根節點計算起，第一層是音樂物件節點，第二層是樂章節點，第三層是片斷節點，第四層是子片斷節點。

Property 2. 對對於每一個節點而言，一個節點會對映到一個「音樂序列（music sequence）」，記為 (b_1, b_2)；表示在音樂物件中，從第 b_1 拍到第 b_2 拍的音樂序列。

Example 1：

參考圖 7 中的樂章節點，對映的音樂序列(1, 40)，表示在音樂物件中，從第 1 拍到第 40 拍的音樂序列。

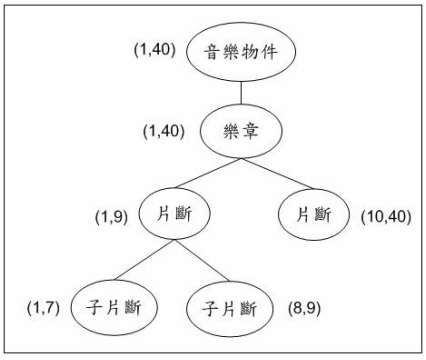

圖7. 節點的音樂序列

Property 3. 對於每一個內部（internal）節點而言，所有子節點的音樂序列，會形成父節點的一個分割（partition）。

Example 2：

同樣地，參考圖 7。對於圖 7 中的樂章節點的音樂序列(1, 40)來說，包含的兩個片斷節點之音樂序列，分別是(1, 9)，(10, 40)，兩個片斷節點都形成樂。

■

關於建構曲式樹的方法，請參考圖6。我們將針對古典樂曲的音樂物件，提出建構曲式樹的方法（heuristics），分述如下：

1. **音樂物件節點**：一首古典樂曲的音樂物件代表一個音樂物件節點。一個音樂物件是由一個 MusicXML document 或是多個 MusicXML document 所儲存。
2. **樂章節點**：對於一首古典樂曲的音樂物件，包含的樂章個數而言，一個樂章代表一個樂章節點。一個樂章就是由一個 MusicXML document 所儲存。
3. **片斷節點**：對於一個樂章節點來說，從樂章中切割出數個片斷特徵。根據休止符的音長，得到片斷特徵。在樂章節點包含的音樂序列中，從序列的第一拍算起，計算至一個音符為休止符，其音長大於等於一拍時，則得到一個音樂序列，將此音樂序列視為一個片斷特徵。若是，計算至最後一個音符，得到的音樂序列，也視為一個片斷特徵。
4. **子片斷節點**：對於一個片斷節點來說，從片斷中切割出數個子片斷特徵。根據音符的音長，得到子片斷特徵。在片斷節點包含的音樂序列中，從序列的第一拍算起，計算至一個音符的音長大於等於兩拍時，則產生一個音樂序列，將此音樂序列視為一個子片斷特徵。若是，計算至最後一個音符，得到的音樂序列，也視為一個子片斷特徵。

Example 3：

我們選取巴哈（J.S. Bach）的「郭得堡變奏曲（Goldberg Variation BWV 988）」的第四變奏之十一個小節來舉例說明，並且，將此音樂片段視為一個音樂物件，參考圖 8。對於圖 8 的音樂片段來說，整個音樂片段就是一個音樂物件，所以對映到圖 9 的音樂物件節點(1, 40)。同時，也就是一個樂章，相對於圖 9 的樂章節點(1, 40)。在樂章節點包含的音樂序列中，從序列的第一拍算起，計算至第九拍為休止符，其音長大於一拍，得到一個音樂序列，將此音樂序列視為一個片斷特徵，相對於圖 9 的第一個片斷節點(1, 9)。在片斷節點包含的音樂序列中，從序列的第一拍算起，計算至第七拍的音長大於等於兩拍時，得到一個音樂序列，將此音樂序

列視為一個子片斷特徵，相對於圖 9 的第一個子片斷節點(1, 7)。

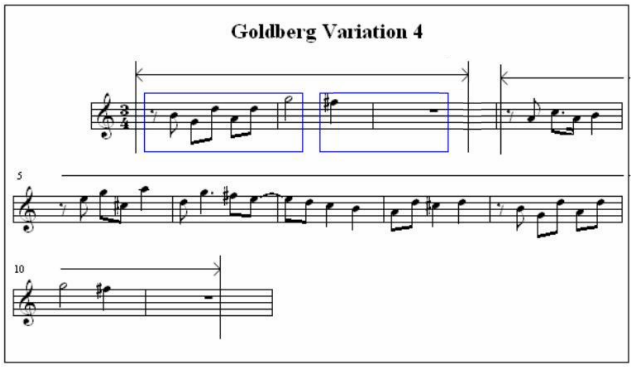

圖8. 巴哈（J.S. Bach）的郭得堡變奏曲第四變奏之十一個小節

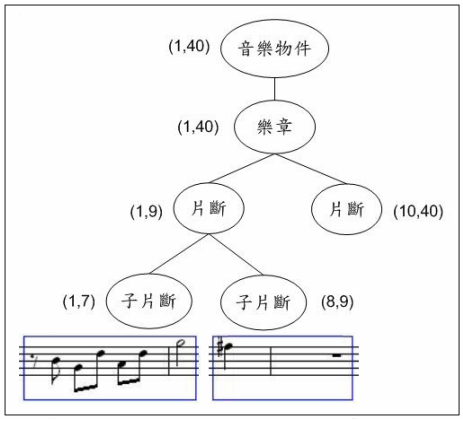

圖9. 曲式樹的示意圖

■

2.4 曲式樹的相似度演算法

在上一個章節中，我們已經介紹了建構曲式樹的方法，利用曲式樹呈現出音樂的結構。接下來，我們介紹曲式樹的不相似度計算法。首先，我們根據節點之間是否重疊，決定節點是否相似。如果重疊，則計算節點之間的不相似度，否則就視為不相關，且不予計算，利用節點的重疊隱含著結構的概念。然後，對於每一個內部節點來說，其不相似度是利用子節點的不相似度得到。因此，我們先介紹 *over*(*u, v*)，將節點代表的音樂序列正規化於 0 至 1 之間後，判斷出節點之間是否重疊，然後，再介紹曲式樹的不相似度計算法。

Definition 2：*over*(*u, v*)節點重疊（overlapping）
給予兩個節點 *u*：(a_1, a_2)與 *v*：(b_1, b_2)，則節點重疊：

$$over(u,v)=\begin{cases}1 & ,\text{if } \dfrac{b_1}{N}\le\dfrac{a_1}{M}<\dfrac{b_2}{N},\text{or } \dfrac{a_1}{M}\le\dfrac{b_1}{N}<\dfrac{a_2}{M}\\ 0 & ,\text{otherwise.}\end{cases}$$

其中，*M* 是包含 *u* 節點之音樂物件節點的音樂序列之個數，*N* 是包含 *v* 節點之音樂物件節點的音樂序列之個數。

Example 4：
如圖 10 所示，給予 t_1 與 t_2 為曲式樹的兩棵子樹，將所有節點的音樂序列正規化於 0 至 1 之間後，如：片斷節點 *a* 的音樂序列(1, 11)之正規化等於(0, 1)，子片斷節點 a_1 的音樂序列(1, 5)之正規化等於(0, 0.45)，同樣地，將片斷節點 *b* 的音樂序列(1, 9)之正規化等於(0, 1)，子片斷節點 b_2 的音樂序列(5, 8)之正規化等於(0.44, 1)。所有的節點都正規化於 0 至 1 之間後，然後，根據節點的位置，得到片斷節點 *a* 與片斷節點 *b* 重疊，子片斷節點 a_1 與 b_1 重疊、a_1 與 b_2 重疊、a_2 與 b_2 重疊、a_3 與 b_2 重疊。

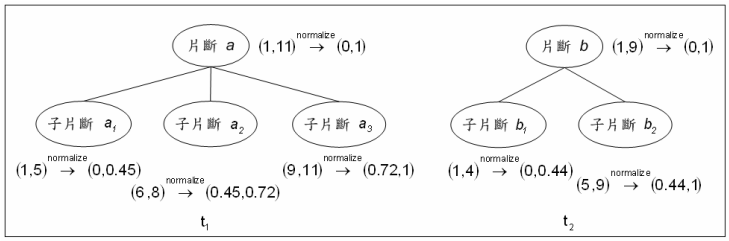

圖10. 曲式樹的兩棵子樹

■

接下來，關於曲式樹的不相似度計算法。如先前所說，如果節點重疊，則計算出節點的不相似度，對於每一個內部節點來說，其不相似度是由子節點的不相似度得到。因此，我們先定義出 $DS_{sub}(a, b)$計算出子片斷節點的不相似度。接下來，$DS_{seg}(a, b)$計算出片斷節點的不相似度，片斷節點的不相似度是由子片斷節點的不相似度得到。同樣地，$DS_{mov}(a, b)$計算出樂章節點的不相似度，$DS_{obj}(a, b)$計算出音樂物件節點的不相似度，兩者的不相似度都是由子節點的不相似度得到。

Definition 3：$DS_{sub}(a, b)$
給予兩個子片斷節點 *a* 與 *b*，則 *a* 與 *b* 的不相似度：

$$DS_{sub}(a,b)=$$

$$(a)\begin{cases}\left(1-\dfrac{v_i\cdot v_j}{\|v_i\|\|v_j\|}\right), v_i \text{ is the pitch histogram of } a\\ \text{N/A} \quad , v_j \text{ is the pitch histogram of } b, \text{if } over(a,b)=1,\\ \quad , \text{otherwise.}\end{cases}$$

$$(b)\begin{cases}\left(1-\dfrac{v_i\cdot v_j}{\|v_i\|\|v_j\|}\right), v_i \text{ is the duration histogram of } a\\ \text{N/A} \quad , v_j \text{ is the duration histogram of } b, \text{if } over(a,b)=1,\\ \quad , \text{otherwise.}\end{cases}$$

對於 $DS_{sub}(a, b)$而言，我們目前提出了兩種計算方式。第一種是將節點內含的音樂序列，根據音符種類(c, d, e, f, g, a, b, r)，得到的音高分佈，計算出兩個節點的不相似度。第二種是將節點內含的音樂序列，根據音長的種類，(全, 2 分, 4 分, 8 分, 16 分, 32 分, 64 分, 休止符)，得到的音長分佈，計算出兩個節點的不相似度。

Example 5：
延續 Example 4，將兩個子片斷節點 a_2 與 b_2，將內含的音樂序列，得到的音高分佈分別為(0, 3, 0, 1, 2, 0, 1, 1)與(3, 3, 0, 1, 0, 1, 0, 4)，由於 a_2 與 b_2 重疊，則：

$$DS_{sub}(a_2, b_2) = \left(1 - \frac{(0,3,0,1,2,0,1,1)\cdot(3,3,0,1,0,1,0,4)}{\|(0,3,0,1,2,0,1,1)\| \ \|(3,3,0,1,0,1,0,4)\|}\right) \times 1$$
$$= 0.42$$

■

Definition 4：$DS_{seg}(a, b)$
給予兩個片斷節點 a 與 b，則 a 與 b 的不相似度：

$$DS_{seg}(a,b) = \begin{cases} \sum_{1\le i\le m, 1\le j\le n} DS_{sub}(a_i, b_j) \Big/ k & \text{, if } over(a,b)=1 \text{ and } over(a_i,b_j)=1 \\ \text{N/A} & \text{, otherwise.} \end{cases}$$

其中，a_i 是 a 的孩子（child），$1 \le i \le m$，m 是 a 的子片斷節點數目，b_j 是 b 的 child，$1 \le j \le n$，n 是 b 的子片斷節點數目，$\sum_{1\le i\le m, 1\le j\le n} over(a_i, b_j) = k$。

Example 6：
延續 Example 4。由於，片斷節點 a 與片斷節點 b 重疊，並且，子片斷節點 a_1 與 b_1 重疊、a_1 與 b_2 重疊、a_2 與 b_2 重疊、a_3 與 b_2 重疊。計算出 a_1 與 b_1 的不相似度為 0.3，a_1 與 b_2 的不相似度為 0.1，a_2 與 b_2 的不相似度為 0.42，a_3 與 b_2 的不相似度為 0.2，則得到兩個片斷節點的不相似度，為 $DS_{seg}(a,b) = (0.3 + 0.1 + 0.42 + 0.2)/4 = 0.255$。

■

同樣地，我們利用計算片斷節點的方法，得到樂章節點與音樂物件節點的不相似度。

Definition 5：$DS_{mov}(a, b)$
給予兩個樂章節點 a 與 b，則 a 與 b 的不相似度：

$$DS_{mov}(a,b) = \begin{cases} \sum_{1\le i\le m, 1\le j\le n} DS_{seg}(a_i, b_j) \Big/ k & \text{, if } over(a,b)=1 \text{ and } over(a_i,b_j)=1 \\ \text{N/A} & \text{, otherwise.} \end{cases}$$

其中，a_i 是 a 的孩子（child），$1 \le i \le m$，m 是 a 的片斷節點數目，b_j 是 b 的 child，$1 \le j \le n$，n 是 b 的片斷節點數目，$\sum_{1\le i\le m, 1\le j\le n} over(a_i, b_j) = k$。

Definition 6：$DS_{obj}(a, b)$
給予兩個音樂物件節點 a 與 b，則 a 與 b 的不相似度：

$$DS_{obj}(a,b) = \begin{cases} \sum_{1\le i\le m, 1\le j\le n} DS_{mov}(a_i, b_j) \Big/ k & \text{, if } over(a,b)=1 \text{ and } over(a_i,b_j)=1 \\ \text{N/A} & \text{, otherwise.} \end{cases}$$

其中，a_i 是 a 的孩子（child），$1 \le i \le m$，m 是 a 的樂章節點數目，b_j 是 b 的 child，$1 \le j \le n$，n 是 b 的樂章節點數目，$\sum_{1\le i\le m, 1\le j\le n} over(a_i, b_j) = k$。

我們利用曲式樹的不相似度計算法，計算出兩棵曲式樹的不相似度，然後，利用階層式分群，得到音樂分類的結果。

2.5 階層式分群

在介紹曲式樹的不相似度計算法之後，我們採用階層式分群（hierarchical clustering）[Theo99]，將音樂物件作分類。初始設定每一個音樂物件分別為一個叢集，在所有的叢集中，利用單一鏈結（single-link）的方法，找到兩個叢集之間的距離為最小，合併成一個新的叢集，重覆至所有的叢集合併成一個叢集則終止。採用階層式分群的原因是由於方法彈性，可以不必事先預設分群的個數。並且，具有視覺化（visualization）的效果。

3. 系統實作

關於我們的系統實作部分，我們撰寫的程式軟體是 Borland C++ Builder 6.0，程式語言是 C++，所使用的電腦配備是 Pentium IV 2.00 GHz，768 MB memory，作業系統是 Windows XP sp1。在我們目前的實作中，剖析 MusicXML document 的模組是來自於 Delphi K. Top 討論區[Delp]。

3.1 系統架構

我們所設計的系統架構，如圖 11 所示，包含下列模組：

- MusicXML document parser：MusicXML document 的剖析。將 MusicXML document 進行剖析之後，並且可以呈現 MusicXML document 的樹狀結構。
- MF-tree module：建構曲式樹。由得到的音樂特徵，將音樂物件建構出曲式樹後，並且可以展示節點的音樂序列之相對位置。

- Clustering module：階層式分群。利用曲式樹的不相似度計算法，得到所有曲式樹之間的不相似度。然後，再利用階層式分群得到我們想要的分群數目。
- Interface module：系統介面。經由實作之後，得到的系統介面實圖。

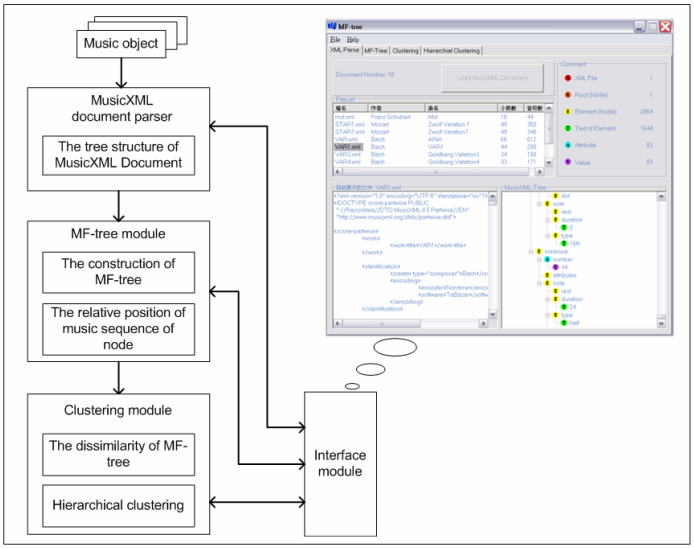

圖11. 系統架構

3.2 系統實作

目前我們系統的實作中，我們將介面分成四個項目：MusicXML document Parser、MF-tree、Clustering、Hierarchical Clustering。

第一個項目是進行 MusicXML document 的剖析，如圖 12 所示。在圖 12 中總共可以分成四個部分，圖 12 中的 A 部分，展示我們讀入的檔案數目、檔名，以及每一個音樂物件的作者、曲名、小節數、音符數等相關資訊。圖 12 中的 B 部分，顯示我們點選的某一個 document 的內容。圖 12 中的 C 部分，是關於點選的 document 之相關資訊，如：元素（element）數目。圖 12 中的 D 部分，展示點選的 document 之樹狀結構。

第二個項目是音樂特徵的擷取，曲式樹的展示，片斷節點位置的展示。如圖 13 所示。在圖 13 中總共可以分成四個部分，圖 13 中的 A 部分，按鈕（Feature Extraction）用來得到音樂特徵。圖 13 中的 B 部分，展示點選的 document，建構成曲式樹之後的相關資料，如：片斷節點的數目。圖 13 中的 C 部分，是曲式樹的呈現。另外，我們將節點加入編號表示。例如：目前展示的曲式樹之片斷節點數目 18 個，則將第一個片斷節點表示為片斷 1，以此類推。若要表示第十三個片斷節點的第一個子片斷節點，則第十三個片斷節點的第一個子片斷節點表示為子片斷 13-1。圖 13 中的 D 部分，是點選曲式樹中的某一個節點，展示節點的音樂序列，在音樂物件上的相對位置。

第三個項目是曲式樹的相似計算，分群結果的呈現，如圖 14 所示。在圖 14 中總共可以分成三個部分，圖 14 中的 A 部分，是可任選兩種方法其中的一種方法，計算出曲式樹的不相似度。圖 14 中的 B 部分，是決定分群的群數，以及執行階層式分群。圖 14 中的 C 部分，展示經由階層式分群，得到分群的結果。

第四個項目是階層式分群的樹狀圖示，如圖 15 所示。在圖 15 中的距離（distance）座標軸，表示兩棵曲式樹之間的不相似度。

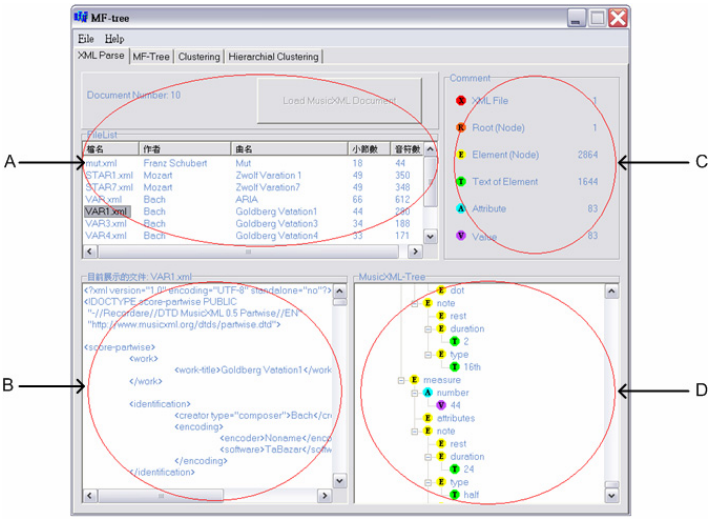

圖12. MusicXML document 的剖析

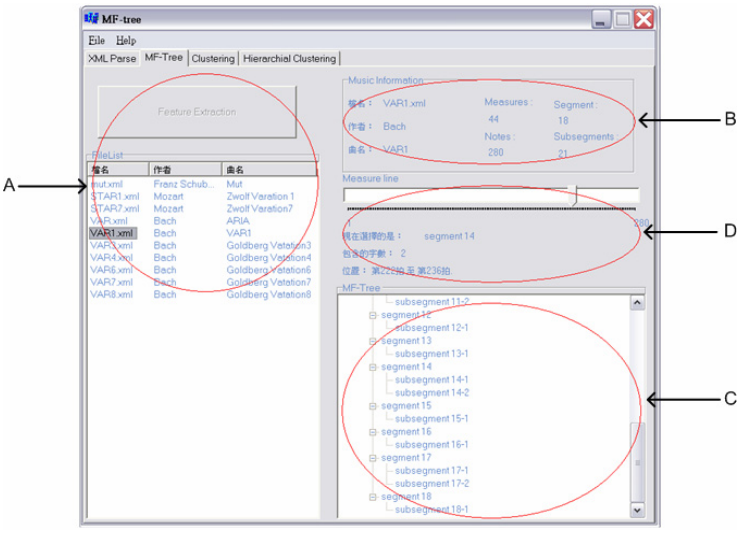

圖13. 曲式樹的建構

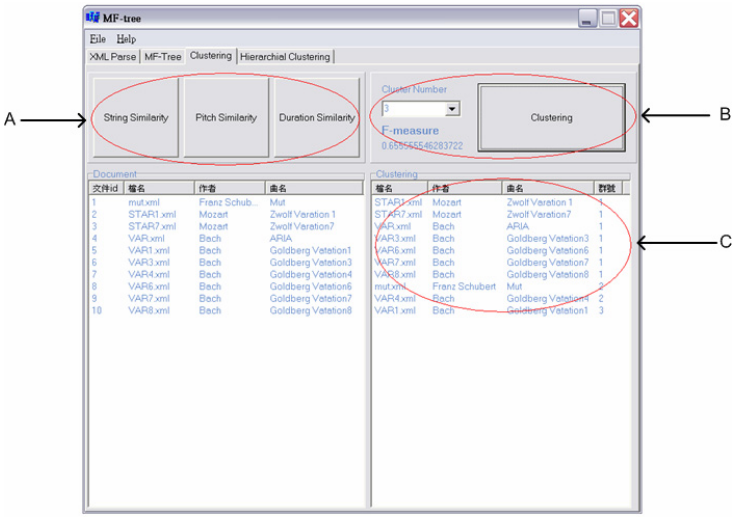

圖14. 分群結果的展示

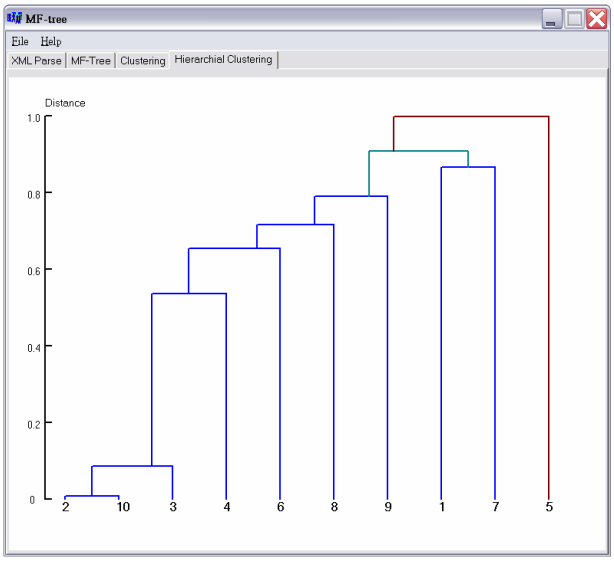

圖15. 階層式分群的樹狀展示

4. 實驗分析

我們提出曲式樹的目標，參考曲式的階層性規則，將音樂物件用樹狀結構呈現出音樂的結構。利用曲式樹的不相似度計算法，計算出曲式樹的不相似度，讓我們在音樂分類上，得到較好的效果。在計算不相似度時，加上了結構的概念，可以提高音樂物件不相似度的準確率，進而提升分群的正確性。我們測試了實驗，測試曲式樹對於音樂物件分群的效果。

4.1 實驗設定

關於資料來源的部分，由於，沒有音樂物件相似的標準（benchmark）。於是，我們將同一首變奏曲中，切割出來的音樂片段視為相似，作為相似的標準，並且，選取出音樂片段具有相似的音高分佈與音長分佈。雖然，從同一首變奏曲中得到的音樂片段，每一個片段讓人聽起來都很相似，但是其包含的音高分佈與音長分佈並不相似。所以，我們將莫札特（Mozart）的「小星星變奏曲（Zwolf Variation）KV 265」，依變奏產生的位置切出音樂片段，同樣地從巴哈（J.S. Bach）的「郭得堡變奏曲（Goldberg Variation）BWV 988」切出音樂片段，加上舒伯特冬之旅（Franz Schubert's Winterreise, D.911）的一首古典樂曲「勇氣（Mut）」，一個音樂片段，總共有十個音樂片段，視為十個音樂物件，每一個音樂物件的節拍數，如表 1 所示。我們是利用評估版的 TaBazar[TaBa]工具，將小星星變奏曲與郭得堡變奏曲等九個音樂物件，轉換成 MusicXML document，剩餘的一個音樂物件的 document 來自於 Recordare[Reco]網站。每一個 document 只儲存音樂物件的第一部之第一樂譜。我們將實驗資料依作者分類，分成三個類別，對於小星星變奏曲兩個音樂片段而言，由於是從同一首曲子中，切出來的兩個音樂片段，所以，將這兩個音樂片段視為一個類別，同樣地，將郭德堡變奏曲的七個音樂片段，視為一個類別，剩餘的一個音樂片段視為一個類別，如表 1。

表1. 實驗資料

作者	歌曲名稱	節拍數
莫札特	小星星變奏曲 變奏一	350
	小星星變奏曲 變奏七	348
巴哈	郭得堡變奏曲 主題	612
	郭得堡變奏曲 變奏一	280
	郭得堡變奏曲 變奏三	188
	郭得堡變奏曲 變奏四	171
	郭得堡變奏曲 變奏六	219
	郭得堡變奏曲 變奏七	215
	郭得堡變奏曲 變奏八	367
舒伯特	夢之後	44

我們利用 F-measure[Larsen99] 評估兩個群組之間分群結果的好壞。假設，專家預先將實驗資料分成 X 類，然後，我們的方法將實驗資料分成 T 群，X 類為我們分群結果的標準。首先，我們先計算每一群與每一類的 Recall 與 Precision，然後，求得每一類與每一群的 F-measure，最後，由全部的（overall）F-measure 的數值來評估分群結果的好壞。其公式如下：

對於每一個 T 群（cluster T）和 X 類（class X），

$$\text{Recall}(X,T)=\frac{n_{ij}}{n_i}\text{，}\quad \text{Precision}(X,T)=\frac{n_{ij}}{n_j}\text{。}$$

n_{ij} = the number of documents of class X in cluster T
n_i = the number of documents of class X
n_j = the number of documents of cluster T

The F-measure of cluster T and class X：

$F(X,T)=\frac{2\times \text{Precision}\ (X,T)\times \text{Recall}\ (X,T)}{\text{Precision}\ (X,T)+\text{Recall}\ (X,T)}$ 。

The overall F-measure $=\sum_X \frac{n_i}{n}\max_T F(X,T)$ ，

其中，n is the number of documents。

由全部的 F-measure 的數值反映出分類結果的品質，數值的範圍從 0 到 1，數值越大顯示品質越好。往後，我們將 overall F-measure 以 F-measure 稱之。

Example 7：假設有五個 documents $D=\{doc1,doc2,doc3,doc4,doc5\}$，已知為兩個類別，$X_1=\{doc1,doc2,doc3\}$ 為一類別，$X_2=\{doc4,doc5\}$ 為另一個類別。我們利用分群演算法，將 documents 分成兩群，$T_1=\{doc1,doc2\}$ 為一群，$T_2=\{doc3,doc4,doc5\}$ 為另一群，用 F-measure 來評估分群的效果，其計算公式如下：

Recall $(X_1,T_1)=\frac{2}{3}$，Precision$(X_1,T_1)=\frac{2}{2}$，

Recall $(X_1,T_2)=\frac{1}{3}$，Precision $(X_1,T_2)=\frac{1}{3}$，

Recall $(X_2,T_1)=0$，Precision $(X_2,T_1)=0$，

Recall $(X_2,T_2)=\frac{2}{2}$，Precision $(X_2,T_2)=\frac{2}{3}$，

$F(X_1,T_1)=\frac{2\times\frac{2}{3}\times\frac{2}{2}}{\frac{2}{3}+\frac{2}{2}}=0.8$，

$F(X_1,T_2)=0.33$，$F(X_2,T_1)=0$，

$F(X_2,T_2)=0.8$，

The overall F-measure

$=\sum_X \frac{n_i}{n}\max_T F(X,T)=\frac{3}{5}\times 0.8+\frac{2}{5}\times 0.8=0.8$

。

■

4.2 實驗結果

我們採用同樣的實驗資料，將每一個音樂物件，擷取出整首音樂音高分佈與音長分佈，將音高分佈與音長分佈利用向量內積，計算音樂物件的相似度。最後，再經由階層式分群得到分群的結果，並且，也利用 F-measure 來評估分群的效果。我們將這兩種方法稱之為 Naïve 的音高長條圖（pitch histogram，以 Naïve-PH 表示）以及 Naïve 的音長長條圖（duration histogram，以 Naive-DH 表示），將這兩種方法得到的分群效果與我們提出的方法比較。

我們根據表 1 將分群數目設定為 3。圖 16 展示我們的方法與 Naïve 方法對於音樂分類的效果。實驗的結果顯示，我們的方法得到的分類效果相對於 Naïve 方法呈現出較好的效果，因此，證實了我們的想法。由於，在計算音樂物件的相似度時，我們將音樂物件利用樹狀結構來表示，根據節點的位置重疊與否，決定節點的是否相似，進而得到曲式樹的相似度。因此，我們不是單純的選取整個音樂物件的音樂特徵來計算，而是利用樹狀結構來表示音樂物件，作為音樂物件的特徵表示。這樣不僅可以提升音樂物件相似度的準確率，同時也增加了音樂分類的效果。另外，我們可以觀察到一件事，由 Naïve 的方法得到的分類結果之數值都是相同。但是，我們得到的分類結果之數值，卻有高低之差。因為在資料不同的音樂特性下，例如：兩首聽起來相似的歌曲，他們的音高分佈是相近的，但是音長分佈是不相近的。利用 Naïve 的方法，是無法區別出這樣的差異，但是，我們利用結構的概念，讓兩首即使是聽起來都相似的歌曲，卻可以辨識出這樣的差異。

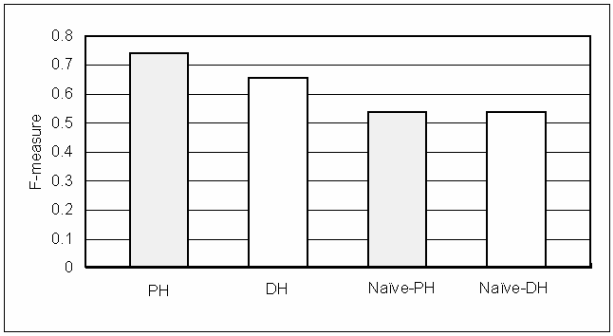

圖16. MF-tree vs. Naïve 的分類效果

5. 結論與未來工作

我們針對 MusicXML 格式的古典音樂物件，參考曲式的階層性規則，建構出一棵具有樂理特徵、樂理結構的曲式樹（MF-tree），並提出曲式樹的不相似度計算法。最後，利用系統實作作為我們成果的展示。

就音樂特徵擷取的概念來說，我們已經不再拘泥於使用字串來表示音樂特徵，而進一步的利用結構來呈現出音樂物件，建構出一棵樹狀結構的曲式樹，並且，也藉由曲式樹得到音樂物件的不相似度。不過，對於特徵的擷取以及不相似度計算的方法，我們認為還有發展的空間。

未來的工作，除了先前提出的音樂特徵，我們可以學習更多的音樂知識理論，有助於我們找到新的音樂特徵。對於曲式樹的不相似度計算法，目前節點的音樂序列之正規化，我們採用線性的方式。不過，我們可以嘗試利用非線性的方式，將節點的音樂序列給予正規化。考慮節點重疊的比例。在系

統方面，延伸目前的系統，提供更多的服務與應用，系統執行的時間。計算子片斷節點不相似度的兩種方法合併成一種，讓使用者多一個選擇。最後，延續我們的實驗，找尋更多的實驗資料一起進行評估，如：以風格作為分類根據的資料，作為我們往後的實驗資料，來驗證我們的方法在音樂分類上的效果。

參考文獻

[Bert04] Bertino, Elisa, Giovanna Guerrini, and Marco Mesoti, "A Matching Algorithm for Measuring the Structural Similarity between An XML Document and A DTD and its Applications," *Information Systems*, Vol. 29, No. 1, March 2004.

[Canf04] Canfora, Gerardo, Luigi Cerulo, and Rita Scognamiglio, "Measuring XML Document Similarity: A Case Study for Evaluating Information Extraction Systems," in *Proc. of 10th IEEE International Symposium on Software Metrics*, 2004.

[Delp]http://delphi.ktop.com.tw/loadfile.asp?TOPICID=13896274&CC=310786

[Hsu01] Hsu, Jia-Lien, Chih-Chin Liu, and Arbee L. P. Chen, "Discovering Nontrivial Repeating Patterns in Music Data," *IEEE Transactions on Multimedia*, Vol. 3, No. 3, September 2001.

[Hsu04] Hsu, Jia-Lien, Arbee L.P. Chen, and Hung-Chen Chen, "Finding Approximate Repeating Patterns form Sequence Data," in *Proc. of 5th International Conference on Music Information Retrieval*, 2004.

[Jone74] Jones, G. T., *Music Theory*, Harper & Row, Publishers, New York, 1974.

[Krum90] Krumhansl, C. L., *Cognitive Foundations of Musical Pitch*, Oxford University Press, New York, 1990.

[Kuo02] Kuo, Fang-Fei, and Man-Kwan Shan, "A Personalized Music Filtering System Based on Melody Style Classification," in *Proc. of IEEE International Conference on Data Mining（ICDM）*, 2002.

[Kuo04] Kuo, Fang-Fei, and Man-Kwan Shan, "Looking for New, Not Known Music Only: Music Retrieval by Melody Style," in *Proc. of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2004.

[Lars99] Larsen, Bjornar, and Chinatsu Aone, "Fast and Effective Text Mining Using Linear-Time," in *Proc. of ACM International Conference on Knowledge Discovery in Data Mining（SIGKDD)*, 1999.

[Lee02] Lee, Mong Li, Liang Huai Yang, Wynne Hsu, and Xia Yang, "XClust: Clustering XML Schemas for Effective Integration," in *Proc. of ACM International Conference on Information and Knowledge Management（CIKM）*, 2002.

[Liu02] Liu, Dan, Nai-Yao Zhang, and Han-Cheng Zhu, "Form Recognition for Johann Strauss's Waltz Centos Based on Music Features," in *Proc. of IEEE International Conference on Machine Learning and Cybernetics*, 2002.

[Miur03] Miura, Takao, and Isamu Shioya, "Similarity among Melodies for Music Information Retrieval," in *Proc. of ACM International Conference on Information and Knowledge Management（CIKM）*, 2003.

[Narm90] Narmour, E., *The Analysis and Cognition of Basic Melodic Structures*, The University of Chinese Press, Chicago, 1990.

[Reco] http://www.recordare.com/default.asp

[Seif03] Seifert, Frank, and Wolfgang Benn, "Semantic Relationship and Identification of Music," in *Proc. of IEEE International Conference WEB Delivering of Music*, 2003.

[Shan02] Shan, Man-Kwan, Fang-Fei Kuo, and Mao-Fu Chen, "Music Style Mining and Classification by Melody," in *Proc. of IEEE International Conference on Multimedia and Expo.（ICME）*, 2002.

[TaBa] http://www.tabazar.de/frame_e.htm

[Theo99] Theodoridis, S., and Koutroubas, K., *Pattern Recognition*, Academic Press, 1999.

[Wai03] Wai, Man Szeto, and Man Hon Wong, "A Stream Segregation Algorithm for Polyphonic Music Databases," in *Proc. of 7th IEEE International Database Engineering and Applications Symposium（IDEAS）*, 2003.

[Wang04] Wang, Lian, David Wai-lok Cheung, Nikos Mamoulis, and Siu-Ming Yiu., "An Efficient and Scalable Algorithm for Clustering XML Documents by Structure," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 1, January 2004.

[Wold96] Wold, E. T. Blum, D. Keislar, and J. Wheaton, "Content-based Classification, Search, and Retrieval of Audio," *IEEE Multimedia Magazine*, Vol. 3, No. 3, Fall 1996.