Int. Computer Symposium, Dec. 15-17, 2004, Taipei, Taiwan.

Rhythm Style Mining of Dance Music

Meng-Fen Chiang, Man-Kwan Shan, Chung-Ching Lan
Department of Computer Science
National Cheng Chi University
Taipei, Taiwan, ROC
{g9309, mkshan, s8910}@cs.nccu.edu.tw

Abstract

Rhythm is one of the basic elements music. Much research on melody or chord had been done in recent years, and there are getting more research that has been observed based on rhythm recently. The aim of our work is to discover dance music style based on rhythm. We consider the timing information of dance music and propose a method to extract and represent rhythm. After mining frequent patterns for each category of music, the result is used to build a classifier and reach our goal, dance music style mining. The performance of our work measured by classification accuracy is up to 95%.

Keywords: Rhythm Extraction, Music Style Mining, Repeating Pattern, Dance Music

1. Introduction

Music is composed of many elements, including melody、rhythm、chord、timbre、tempo, etc. Humans can identify these elements by perception. While digital music is becoming one of data types propagate on daily life, there are much demand to analyze the elements of music by computers. And music content analysis becomes a popular subject to work on.

A piece of music can be represented as a melody sequence、beat sequence, etc. There is much research based on melody, but only a little research about rhythm are examined. We believe that there must be much potential information could be found by analyzing timing information of music.

The term beat refers to regularly recurring pulses within a given period of time. Any single note value can serve as the designated beat unit of a composition. Meter is the grouping of beats into units known as measures. Rhythm is a temporal pattern played against a background of beat units. Sometimes it corresponds to the beat, pulse, or stress, but sometimes it may conflict with them. Rhythm is made of a group of note. If a note is viewed as a music event, then music could be represented as an event sequence. Beat and rhythm are closely related. Music could be made of different duration of notes in different order. These different structures make the strong /weak、short/long duration of beat. A beat sequence with strong/weak and short/long duration of beats makes rhythm.

The aim of this paper is to extract rhythms from dance music and find out the syntactic description of music style based on extracted rhythm. As we know, rhythm has the characteristic that it always occurs recursively in music. Rhythm associated with dance or instrumental forms are most often repetitive, recurrent, and symmetrical. [1] We collected dance music MIDI files and investigated the rhythm extraction method to compute and discover repeating patterns that lately used to discover the music style.

Generally speaking, if a piece of music could be used to dance, then it is dance music. We restrict the dance music here to ballroom dance music. For instance, Salsa、Samba、Tango, etc are the objects we work on. Dance music of the same category shares the same rhythm style. For example, the rhythm style of waltz is of the form 100010001000 where 1 means note onset and 0 means the duration the note lasts.

Our work makes the following applications:

(1) It can be applied in personalized content-based music retrieval (CBMR) system based on users' preference. It's a system that can discover users' preferred music style by examining and learning users' retrieval activities.

(2) It can be used automatic composition system to edit music through selecting rhythm patterns. Therefore, users can arbitrarily compose music by selecting patterns that can be standard dance music patterns of each style or users' preferred music style pattern produced by personalized CBMR system.

2. Related work

Simon Dixon [3] classified dance music by discovered metrical hierarchy and timing pattern. One of the steps is to extract periodicities. Because sources are wave files, Dixon has to detect the onset before finding periodicities and then calculate inter-onset intervals for further clustering. Another method of detecting periodicities uses autocorrelation on the amplitude envelopes of band-limited versions of signal. The relationships between periodicities are then used to find the metrical hierarchy and estimate the tempo at a few levels of the hierarchy. The last step is to predict the style of music by periodicities、tempo、meter and the distribution of periodicities. One thing needs to know is that periodicities give information about the metrical structure of the music, but not the rhythmic structure.

Benoit Meudic [2] reported an automatic meter extraction system based on auto-correlation coefficients. The input format for music is MIDI. The beat sequence and its occurrences in the music are assumed known. The algorithm generates a set of possible metrical groupings. Meter is a grouping that repeats at least one time in a music sequence. Groupings can be integrated into higher level grouping. Thus, the metrical hierarchy also can be generated in Meudic's experiment. For each metrical level, the algorithm generates a list of groupings and the best groupings are selected according to its frequency.

3. Method

We first introduce MIDI briefly before presenting our method. MIDI is digital music data. It consists of many tracks and each track indicates a particular instrument. Each track contains a sequence of notes. Every track records notes' information such as onset time, ending time, volume, pitch, and the track that a note belongs to. (Figure. 1)

tick 0: ch. 2: note On G5 vel.: 125
tick 120: ch. 2: note On G5 vel.: 0
tick 120: ch. 2: note On E5 vel.: 127
tick 240: ch. 2: note On E5 vel.: 0
tick 240: ch. 2: note On E5 vel.: 126
tick 480: ch. 2: note On E5 vel.: 0

Figure 1. An example of information of a note sequence

Our method can be divided into five steps. The first, second and third steps are performed for each music. The first step is to extract useful information that is sufficient to compute notes' duration from parsed MIDI data and then transform into beat string. The second one is to transform the beat string into measure string according to its meter. The third step is to discover repeating patterns from measure string. The repeating patterns correspond to rhythm. Only the high frequency patterns are selected. The correlative matrix algorithm [4] is utilized to find repeating patterns.

In the fourth step, frequent pattern mining technique is employed first to derive the common properties and the interesting hidden relationships between rhythm and styles from dance music of the same category. The frequent patterns indicate the common properties of the music objects belong to the same style. The concept of frequent itemset in the association rule mining [6] is utilized in this paper to discover the frequent patterns of music in the same category. However, it is not enough to discriminate one style from others only by the frequent patterns. In generally, human recognize a music style not only by the characteristics of itself, but also by the discrimination between its style and others. Therefore, after mining frequent patterns from each category, the fifth step utilizes the classification algorithm to generate discrimination rules and the rules can be regarded as the rhythm style pattern. The following shows the flowchart of our proposed method.

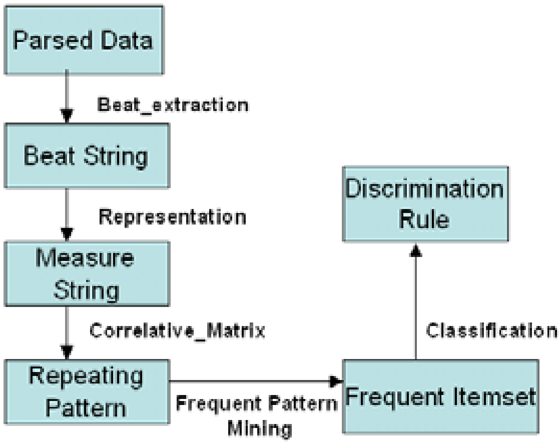


Figure 2. The flowchart of the proposed method

Note that before extracting timing information from MIDI files, we have to parse MIDI files with our MIDI parser. Notes' onset time and duration are the information we need for the first step, beat extraction.

3.1 Beat extraction

Rhythm related instrument can be found in MIDI instrument index file. For dance music, percussion instruments are usually rhythm related instrument that are set to channel 10 by MIDI composer. Therefore, we have to extract all notes with channel 10. Figure.3 shows all the percussion instruments that are set to channel 10 in one of the salsa style music, elraton.

Chn	Name
10	bongo
10	clave
10	congas
10	maracas
10	campana
10	timbal

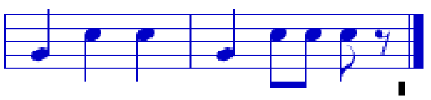
Figure 3. All instruments of channel 10 in elraton (snapshot from Cakewalk)

We extract note onset time and ending time to calculate beat duration. We choose shortest note as the unit of time slot that we will use afterward. Beat string is the string that consists of number 1 and 0 representing note onset and numbers of time slot within duration respectively. If we use number 1 stands for note onset time and number 0 stands for every time slot within note duration, every note can be represented as a string beginning with number 1. For instance, slot unit is set sixteenth note. Then, a quarter note can be represented as 1000, totally 4 slot unit long with 1 as onset time and 0 as reset of the string. See Figure. 4, every note is translated into beat string with slot unit sixteenth note.

note	beat string
whole note	1000000000000000
half note	10000000
quarter note	1000
eighth note	10
sixteenth note	1

Figure 4. Each type of note's beat string with slot unit sixteenth note

Take Figure. 5 as an example, we try to represent a piece of music with 2 measures. There are 7 notes, and its corresponding beat string with eighth note time slot is 10101010111.



10 10 10 10 1 1 1 0

Figure 5. Beat string of the corresponding piece of music

To translate a piece of music into beat string, precise onset time and the duration of notes is a key point. However, most of MIDI files are the results of music performance and therefore are not consistent with the corresponding original music score. It makes the slight difference on onset time and duration, and so does the error while translating beat string. Therefore, appropriate alignment on onset time and duration is needed.

After the time slot unit is determined, we have to tune note onset time and then round the difference between the onset time of performance and the original score. If the difference is beyond the half of time slot, it's necessary to adjust the onset time.

Otherwise, keep the original value and tune the duration time afterward. We divided the duration into parts with size of slot unit. Adjust duration if the last part is beyond the half of slot, otherwise truncate this part. Figure.6 gives an example. Given the onset and ending time of each note from MIDI parser, it is not necessary to adjust the note beginning from tick 1248 to 1254. On the other hand, we have to adjust the onset time of the note from tick 1264 to 1266. This is because that tick 1264 is not on the score time tick 1260 or tick 1266. The corresponding duration is then shifted to right which begins at tick 1266. Duration is divided exactly by time slot, 6 ticks, so the time of this duration remains unchanged.

tick	channel	note	pitch
1248	10	On	A5
1254	10	Off	A5
1264	10	On	A5
1270	10	Off	A5

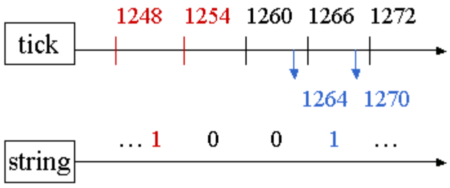


Figure 6. Align onset time and the duration with time slot 6 ticks

3.2 Measure string representation

After we get the beat string from the first step, we start to transform the beat string into measure string. Measure string is the string that consists of sections of beat string with string length corresponding to each measure.

The difference of these two is that the time slot of beat string can be set arbitrarily, but slot is set to measure in measure string.

We divide beat string according to measure length defined by meter and get numbers of beat string sections. Table 1 gives an example. Given the beat string "B = 101010101110" of the corresponding piece of music from Figure.5, string B can be divided into two measure strings denoted as m1 and m2.

Table 1. Measure string produced from the beat string of Figure.5

Original beat string	101010101110	
Corresponding measure string	101010	(m1)
	101110	(m2)

3.3 Finding repeating patterns

A repeating pattern is a sequence of notes which appears more than once in a piece of music. For example, the repeating patterns of string "ABABAB" are "A", "B", "AB", "BA", "ABA", "BAB", and "ABAB". According to matching similarity, there are two types of repeating patterns, exact and approximate repeating patterns respectively. Approximate repeating pattern is found by providing fault tolerance string matching approach. The approach allows some errors such as dropout errors, insertion errors, and transposition error.

We aim at finding exact repeating pattern in our work. There are few approaches for finding repeating patterns such as suffix tree approach, correlative matrix [4] and string-join approach [8]. The suffix tree approach requires more storage space and execution time than the other two. The string-join approach performs better than correlative matrix approach while discovering longer repeating patterns. Since rhythm is shorter repeating patterns, there is no apparent difference while finding the high frequency shorter repeating patterns. Therefore we adopt the method, correlative matrix, to discover repeating patterns.

The correlative matrix approach has three steps. First, the approach generates the correlative matrix for the discovery of repeating pattern based on the dynamic programming technique. Correlative matrix makes repeating pattern finding more efficient that it can keep the intermediate results of substring matching. It saves time to mark the length of repeating substring while matching patterns. Table 2 is the generated correlative matrix for repeating pattern finding of a music segment *S* ="ABABABABCDDEC" where each character represents a measure string. Let s[i] denote the i^{th} measure sting of this music segment. Matching s[i] and s[j] are sequentially performed where i $\neq$j and i>j. If s[i] equals s[j], the cell in the i^{th} row and j^{th} column is set to 1. Besides, if the s[i+1] row equals s[j+1], then the cell in the $i+1^{th}$ row and $j+1^{th}$ column is set to two. Namely, the number in the cell indicates the length of a pattern.

Table 2. The correlative matrix

	A	B	A	B	A	B	A	B	C	D	D	E	C
A	-		1		1		1						
B		-		2		2		2					
A			-		3		3						
B				-		4		4					
A					-		5						
B						-		6					
A							-						
B								-					
C									-				1
D										-	1		
D											-		
E												-	
C													-

After the generation of the correlative matrix, the second step is to find potential repeating patterns by ignoring the trivial patterns. A potential repeating pattern *P* is trivial if there exists another potential repeating pattern which is the super-string of *P* with the same repeating frequency. The last step is to discover the repeating patterns from the potential repeating pattern. Table 3 list all of the repeating patterns found in string *S*.

Table 3. All of the repeating patterns found in *S*

Repeating	Pattern Length	Repeating Pattern
4	1	B
2	1	C
2	1	D
4	2	AB
3	3	BAB
3	4	ABAB
2	5	BABAB
2	6	ABABAB

Even with the elimination of the trivial repeating patterns, there still exist numerous repeating patterns for each music. However, the rhythmic pattern of music is the recurrent pattern with high frequency. Therefore, only the top high frequency repeating patterns are retained for each music.

3.4 Frequent pattern mining

After finding the repeating patterns, the next step of rhythm style mining is to mine the common characteristics of each music category sharing the same style using the extracted repeating patterns. We utilized the concept of frequent itemset in the association rule mining [7] in order to obtain the interesting relationships between rhythm and music style. Therefore, each item corresponds to a repeating pattern, and each music corresponds to an itemset. An itemset is frequent if over *s%* (minimum support) of training music that contain this itemset in whole database. For instance, if the itemset {A, B}, where A, B denote the repeating pattern 100010001000, 1000010010000100100000100 respectively, of waltz music is frequent, that means a great part of waltz music consist of these two rhythm patterns. We implement the Apriori algorithm [7] to find the frequent itemsets.

3.5 Classification

As stated in the beginning of this section, after the mining of common characteristics, the last step of rhythm style mining is to discover the discrimination rules using classification technique for rhythm style mining. Our classification method adopts the concept of associative classification [5]. In associative classification, a rule is of the form $l \Rightarrow y$, where $l \in \cup_k L_k$, *l* is a frequent itemset and *y* is a class. A rule $l \Rightarrow y$ with confidence *c* if *c%* of training samples in dataset that satisfy *l* belong to class *y*. A training sample conforms to *l* if its corresponding repeating patterns contain *l*. The rule $l \Rightarrow y$ has support *s* if *s%* of the training samples in dataset satisfy *l* and belong to class *y*.

The classification algorithm construct a classifier several passes over the training data using heuristic method. The classifier is of the form $< r_1, r_2, ..., r_n, default\_class >$, where rules r_i are ranked by the confidence and support. The test data are classified by the first rule that sequencially matched. If there are no rules satisfying the test data, the test data will be classified according to the default_class.

We have investigated the melody style classification method and proposed a classification algorithm Single-Type Uniform Support Classification (STUS) [7] which is the modification of the associative classification algorithm. To improve the performance of STUS, we have also developed use a new classification algorithm – Single-type Variant-Support Classification (STVS) [7]. The uniform support used in associative classification algorithm may be not appropriate for all cases in the classification. For example, if the style of composers is diverse, there may be more number of rules but with lower supports. Therefore, we apply STVS to build a classifier that contains rules of different types of patterns. Figure.7 shows the algorithm.

Algorithm *Single-Type-Variant-Supports-Classification*
Input: music database *MD*,
candidates of *min_sup MS*,
number of folds v
Output: *Classifier*
1. divide training data of each category *y* into *v* subsets $T_{y,k}$
2. for each combination of *min_sups* and multiple patterns of all categories do
3. for k = 1 to v do
4. for each category *y* do
5. $training\_set_y = \bigcup_{i \neq k} T_{y,i}$
6. $validation\_set_y = T_{y,k}$
7. mine frequent itemsets from $training\_set_y$
8. for each type of patterns do
9. calculate confidence of frequent patterns
10. *Classifier* = ***Training-Classifier***
11. classify each $validation\_set_y$ by *Classifier* and store the accuracy a_k
12. accuracy of all combinations $= \sum a_k / 5$

Figure 7. STVS Classification Algorithm [7]

Figure.8 gives an example of a 2-way STVS classifier for classification of two music categories X and Y. If a testing music consists of the repeating pattern with measure string 100000001000000010000000, then the testing file is classified to class Y, otherwise to default class X.

100000001000000010000000 $\rightarrow$ class Y
Default_Class: class X

Figure 8. A two-way STVS Classifier

4. Experiment

4.1 Experiment set up

We collected three categories of MIDI files from the Internet. They are three categories of music, Tangle, Waltz, and Bossa Nova. Each category of the MIDI files were retrieved from the top 20 web sites returned from google with key words "midi" and the category name. Each category remains 45 to 50 files after cleaning files incompatible with our MIDI parser.

The environment for executing our experiment is listed in Table 4.

Table 4. Environment for experiment

Server	Avatar
CPU	Dual CPU 2.4GHz
RAM	2047 MB
Virtual Memory	4G MB

4.2 Performance analysis

The performance of our work is measured by the accuracy of the 2-way classification. We use 5-fold cross-validation to measure the accuracy of the classification method. The accuracy is *A* if *A%* of the test songs are classified to the correct category which we've known from the text description while we collect MIDI files. The testing files of each category are randomly partitioned into five equal-sized mutually exclusive subsets (folds). Training and testing are performed five passes. One of the folds is selected as the test set while the other four are collected to derive the classifier in each pass. The average accuracy calculated from the five tests is then used to evaluate the performance of classification.

Factors that affect the performance include minimum support, and the number of retained repeating patterns of each music. We made the experiments by considering the effect of these factors. Table 5 shows the result where X, Y, Z denotes the category of tango, waltz and the bossa nova songs respectively.

Table 5. Performance of the experiment

Minimum Support	Accuracy					
	Number of retained repeating patterns = 3			Number of retained repeating patterns = 5		
	X-Y	X-Z	Y-Z	X–Y	X–Z	Y–Z
10-10%	85.33%	**91.11%**	**95.56%**	87.46%	83.3%	**95.56%**
10-20%	**87.33%**	88.9%	93.33%	90%	83.3%	83.3%
10-30%	**87.33%**	88.9%	93.33%	**93.3%**	83.3%	93.33%
20-10%	77.22%	**91.11%**	**95.56%**	84%	**92.2%**	**95.56%**
20-20%	69.89%	87.78%	93.33%	72.1%	90%	93.33%
20-30%	69.89%	87.78%	93.33%	72.1%	90%	93.33%
30-10%	75%	87.78%	**95.56%**	84%	87.78%	**95.56%**
30-20%	67%	85.56%	93.33%	67%	85.56%	93.33%
30-30%	67%	85.56%	93.33%	67%	85.56%	93.33%

The accuracy reaches about 95%. The accuracy in boldface shows the best combination of minimum support combination for each 2-way classification. We found that more number of retained repeating patterns produces better performance. Moreover, we observed that the style each pair of category is distinct between each other. This is also consistent with our intuition. Since waltz is triple measure dance music different from other two quadruple time dance music.

Moreover, by inspecting the classifier, bossa nova has more tiny beats uniformly distributed in each measure, but tango doesn't. Figure 9 shows part of frequent patterns of each category with pattern size corresponds to one measure..

10000000**1**0000000**1**0000000**1**000**1**0**1**0 (tango)
10000000**1**0000000**1**0000000 (waltz)
1000**1**000**1**000**1**000**1**000**1**000**1**000**1**000 (bossa nova)

Figure 9. One of the frequent patterns of each category

5. Conclusions

In this paper we proposed a method to discover the rhythm style from music. The information we needed is parsed from the MIDI files collected from the Internet. Beat strings are generated according to onset time and duration of notes that parsed forward. During beat extraction, we align onset time and duration occurred in the MIDI files. To transform the beat string into measure strings is performed. Afterward, the repeating patterns occur in music can be discovered by using correlative matrix to extract repeating patterns in measure string data. The top high frequency repeating patterns are retained for each music. After discovering the repeating patterns, the next step of rhythm style mining is to mine the common characteristics of each music category sharing the same style using the extracted repeating patterns. The last step of rhythm style mining is to discover the discrimination rules using classification technique for rhythm style mining. The performance of the experiment measured by classification accuracy is up to 95%.

The process of rhythm extraction could further applied for the development of application system. For instance, a content-based music retrieval system applied in P2P search system, such that it can search music not only by meta data, but also by rhythm. Besides, we can create an automatic composition system to edit music given a specific rhythm patterns. A personalized recommendation (personalized CBMR) system that searching music according to rhythms of the music that lately listened by user.

So far we just consider onset and duration of a note, but haven't investigated the effect of strong and weak beat. The performance of rhythm style mining may be raised with consideration of strong and weak beat.

References

[1] L. Stein, *Anthology of Musical Forms*, Warner Bros, 1962

[2] B. Meudic, "Automatic Meter Extraction from MIDI Files," Proceedings of the 3rd Journées d'Informatique Musicale, 2002.

[3] S. Dixon, E. Pampalk, G. Widmer, "Classification of Dance Music by Periodicity Patterns," Proceedings of the 4th International Conference on Music Information Retrieval, 2003.

[4] J. L. Hsu, C. C. Liu, and Arbee L. P. Chen,, "Efficient Repeating Pattern Finding in Music Databases," Proceedings of International Conference on Information and Knowledge Management, 1998.

[5] B. Liu, W. Hsu, and Y. Ma, "Integrating Classification and Association Rule Mining," Proceedings of ACM International Conference on Knowledge Discovery and Data Mining KDD, 1998.

[6] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proceedings of International Conference on Data Engineering, 1995.

[7] F. F. Kuo, and M. K. Shan, "A Personalized Music Filtering System Based on Melody Style Classification," Proceedings of International Conference on Data Mining, 2002.

[8] C. C. Liu, J. L. Hsu, and Arebee L. P. Chen, "Efficient Theme and non-Trivial Repeating Pattern Discovering in Music Databases," Proceedings of International Conference on Data Engineering, 1999.