Melody Extraction on MIDI Music Files

Giyasettin OZCAN
Dokuz Eylul University
Dept of Computer Eng.
ozcan@cs.deu.edu.tr

Cihan ISIKHAN
Dokuz Eylul University
Dept of Musicology
cihan.isikhan@cs.deu.edu.tr

Adil ALPKOCAK
Dokuz Eylul University
Dept of Computer Eng.
alpkocak@cs.deu.edu.tr

Abstract

In this study, we propose a new approach to extract monophonic melody from MIDI files and provide a comparison of existing methods. Our approach is based on the elimination of MIDI channels those do not contain melodic information. First, MIDI channels are clustered depending on pitch histogram. Afterwards, a channel is selected from each cluster as representative and remaining channels and their notes are removed. Finally, Skyline algorithm is applied on the modified MIDI set to ensure accuracy of monophonic melody. We evaluated our approach within a test bed of MIDI files, composed of variable music styles. Both our approach and the results from experiments are presented in detail.

Key words: Melody Extraction, pitch histogram, Music Information Retrieval

1. Introduction

As digital music files appears through Internet, Content-based Music Information Retrieval (MIR) has been receiving much attention by research groups. The ultimate goal of content based MIR is to be able to retrieve relevant data from a music database using a melody fragment as a query. One of the most crucial problems of this area is to extract and represent the melodic content of the music data. Eliminating accompany notes/channels from melodic representation form is somewhat similar to stop-word removal as in text based information retrieval and it is not a trivial task.

Polyphonic nature of the music permits simultaneous sound of notes. As a result, polyphonic music retrieval is computationally expensive. In contrast, monophonic music consists of linear note sequence and it is more appropriate for music retrieval. Melody Extraction is a process, which generates monophonic equivalent of polyphonic files. Hence, output of Melody Extraction takes less space in databases by containing only genuine part of the music. Moreover, retrieval on generated monophonic files can be executed in a reasonable time.

Uitdenbogerd and Zobel proposed four different melody extraction techniques [12][13]. Their first technique, called Skyline, collects all the notes of a MIDI file into single MIDI channel. Then, algorithm simply selects highest pitch line of the note sequence as melody. If necessary, Skyline also modifies note durations as well. Their last three algorithms were based on the selection of the best MIDI channel contributing melody. As a result remaining channels are assumed to be accompaniment and discarded. Some cognitive criterions such as pitch average or entropy of a channel are used in channel selection process. However, all four algorithms have drawbacks; if melody is located on lowest pitch notes, or if melody is distributed on multiple channels (i.e., instruments in MIDI case) algorithms may produce very poor performance and fail.

Chai proposed a revision on Skyline Algorithm [2] and pointed out that original note durations should be kept on. In her approach, low-pitch frequency notes are tested first and notes causing relatively high contribution to polyphony are eliminated. Later on, Shan claimed that volume of the notes may disclose the location of melodic content and presented a new melody extraction algorithm [3]. However, none of the modifications solved the above mentioned problem.

Main contribution of this study is to introduce a new approach of melody extraction from MIDI files, which is independent of how melody is located throughout the MIDI file. The main idea of our approach is selecting all the channels involving
melody. This new approach is called as Best-\(k\) channel since it selects \(k\) channels, where \(k \geq 1\), for melody extraction. For \(k\) channel selection, we propose two methods: Clustering and Top Rank. In clustering method, first, MIDI channels are clustered depending on the pitch class histogram. Afterwards, a single channel is selected from each cluster as representative. In Top Rank approach, \(k\)-channels which are the most similar to entire MIDI are selected and remaining channels are eliminated. Finally, Skyline algorithm is applied to get the monophonic melody as output.

In order to evaluate our approach and provide a comparison of existing methods, we have compiled MIDI files with different melodic styles and layouts (i.e., Melodic content of the music might be distributed among channels or located both higher and lower pitch notes etc.).

The remainder of the paper is organized as follows. Section 2 starts with some basic definitions and notations we used throughout the paper and gives a brief look at the previous works on this subject. Section 3 presents the details of our approach. In Section 4, evaluations of the melody extraction algorithms are presented and test results are discussed. Finally, Section 5 concludes the paper and gives a look to the further studies on this subject.

2. Definitions and Related Works

Historically, the music has been represented, mainly, by three different formats: written score, recorded performance and, recently, the MIDI format. All three music formats have different characteristics and music in digital format mostly appears as recorded performance (i.e, in MP3 or wave format). Primarily, MIDI format is developed for exchanging musical data between electronic instruments. Later on, MIDI format become so popular. In this study, we use MIDI file format, because MIDI files are readily available and it is almost digital replacement of written score.

Assume that \(M\) is a MIDI file composed of channels. Formally,
\[
M = \{ c_1, c_2, \ldots, c_i \} \text{ where } 1 \leq i \leq 16.
\]  
(1)

Each channel, \(c_i\), is a set, containing \(k\) notes. Mathematically,
\[
c_i = \{ n_{i1}, n_{i2}, \ldots, n_{ik} \}
\]  
(2)

For our case, each arbitrary note, \(n_{ij}\), has three important properties: pitch, onset and offset time, \(p_{ij}\), \(s_{ij}\) and \(e_{ij}\), respectively. Formally, we define a note as a set as follows:
\[
n_{ij} = \{ p_{ij}, s_{ij}, e_{ij} \} \text{ where } 1 \leq p_{ij} \leq 128
\]  
(3)

MIDI notes are sorted by onset time. Therefore,
\[
\forall n_{ij}, n_{i(j+1)} \in c_i \text{ ; } s_{ij} \leq s_{i(j+1)}.
\]  
(4)

However, music does not expose constraints for the offset time. If offset time of a note, \(e_{ij}\), is greater than onset time of successive note, then polyphony will occur. Formal definition of polyphony is as follows:
\[
\exists n_{ij}, n_{i(j+1)} \in c_i \text{ ; } e_{ij} > s_{i(j+1)}
\]  
(5)

In literature, one of the major approaches to extract melody from polyphonic files is the Skyline Algorithm [12]. Principally, it collects all notes into one channel and selects the highest pitch notes. Skyline Algorithm is shown in Algorithm 1. In case of multiple notes have same onset time, note with maximum pitch frequency is kept, whereas rest of notes are eliminated (line 4-10). Secondly, it may shorten the durations of notes. In line 11-12, when a new note occurs with a different onset time, duration of the existing note will be shortened.

Although Skyline yielded good results, three critics can be made. Firstly, manipulating the note durations may change melody. Secondly, collecting all notes into one channel removes the silent intervals which may carry on hidden melody. This problem can be seen Figure 1. Although melody is maintained by primary channel, notes from secondary channels takes place and removes melodic rests coming from primary channel. Figure 1 implies that elimination of secondary channel improve music retrieval. Third critic said that accompany notes might have high frequency.

**Algorithm 1**: Skyline

begin  
1. \(j := 1; \ i = 1;\)
2. for each \(n_{ij} \in M\) do  
3. \(k := j + 1\)
4. while \((s_{ij} \neq s_{ik})\) do  
5. \(\text{if } (p_{ij} < p_{ik})\)  
6. \(\text{eliminate } p_{ij}\)
7. \(j := k\)
8. else  
9. \(\text{eliminate } p_{ik}\)
10. \(k := k + 1\)
11. \(\text{if } e_{ij} > s_{ik} \text{ then } e_{ij} = s_{ik}\)
12. \(j := k\)
13. end for  
end
Keeping the original durations of notes was proposed as a recovery of Skyline by Chai and called as Revised Skyline. Starting from the note, \( n_{ij} \), with minimum \( p_{ij} \), each note, is tested for its contribution to polyphony. If at least 50% of the duration between \( s_{ij} \) and \( e_{ij} \) is polyphonic, then \( n_{ij} \), will be eliminated from the set. Since notes with high pitches are tested lately, rarely they contribute to polyphony and stay in the final set.

2.1 Algorithms based on Channel Selection

First attempt to channel selection has been done by Ghias et al. [6]. They showed that percussion channel never contributes to melody. As a result, elimination of percussion channel not only enhances the relevancy of the search, but also speeds up the retrieval time. Furthermore, Uitdenbogerd and Zobel proposed methods to select best MIDI channel representing melody [12]. Firstly, they obtained Skyline output of each MIDI channel and they computed average pitch frequency and entropy of each channel. At the end, the channel having the maximum pitch average or entropy was selected as the melody channel. In addition, Shan et al. showed melody notes have high volume [3].

![Figure 1: First four measures of Alla Turka, W.A. Mozart, in two MIDI channels. a) Primary channel contains melody whereas secondary channel includes accompany. b) - Manipulated notes after Skyline Algorithm, c) - Notes sequence after removal of secondary channel.](image)

Experiments show that success of channel selection algorithms highly depends on the style of the music. For example, if perceived melody is distributed among channels, selection of one melody channel will lead to loss of melodic information. Consequently, the perfect solution lies at the capability of selecting all channels involving melody.

3. Best-\(k\) Channel Algorithm

If melody is located in one channel, existing algorithms based on selecting only one melody channel, such as Top channel algorithm, may perform a satisfactory result. However, if melody is distributed among multiple channels, it will be almost impossible to extract the melody properly. Therefore, we propose another approach for melody extraction which is based on selecting all the channels involving melody. This new approach is called as Best-\(k\) channel since it selects \(k\) channels, where \(k \geq 1\), for melody extraction. Depending on the MIDI file, approach estimates the best \(k\) value.

In order to determine a proper \(k\) value, we used the histogram features of each MIDI channel. As a result of pitch histogram, we computed the dissimilarity scores between each channel and entire set and consequently clustered MIDI channels. We present that \(k\) can be equal to the total number of clusters. Since each MIDI and its channels have their own histogram properties, total selected channels, \(k\), is not deterministic.

3.1 Melody Channel Selection Process

We implement three preliminary operations before computing pitch histogram. First, we eliminate percussion channel, \(c_{10}\) [6]. Second, we apply Skyline Algorithm to all channels [12][13]. Third, we represent MIDI notes by 12 semitones. In order to achieve this, we computed modulo 12 equivalents of pitches [9]. As a result, pitch histogram of channels become a point in 12-dimensional space.

For selecting best-\(k\) melody channels, we propose two methods: Clustering and Top Rank. Before further discussion of how our channel selection proposals work, let us give formal definitions to clarify it. Let \(h_i\) is the pitch class histogram of \(c_i\), then histogram set \(H\) will be:

\[
H = \{ h_1, h_2, \ldots, h_{16} \} \quad (6)
\]

Since we define the music with 12 semitones, each \(h_i\) consist of 12 dimensions as well. Formally,

\[
h_i = \{ h_{i1}, h_{i2}, \ldots, h_{i12} \} \quad (7)
\]

Assuming that \(T\) is the number of channels which contains at least one note, then normalized value of \(i^{th}\) dimension, \(\overline{h_i}\), is defined as follows:
\[
\overline{h}_i = \frac{\sum_{k=1}^{16} h_{ki}}{T} \quad (8)
\]
Correspondingly, average histogram of entire channels, \( \overline{h} \) will be:
\[
\overline{h} = \{\overline{h}_1, \overline{h}_2, ..., \overline{h}_{128}\} \quad (9)
\]
In our approach, we compute dissimilarity between each channel and \( \overline{h}_i \). Let assume that \( D(h_i, h_j) \) is a function computes the Euclidian distance/dissimilarity between \( h_i \) and \( h_j \). Then,
\[
d_i = D(h_i, h_{A}) \quad (10)
\]
For our purpose, we keep channel distances in a new set. Formally,
\[
DS = \{d_1, d_2, ..., d_{16}\} \quad (11)
\]
In summary, shorter \( d_i \) leads to more similarity between a channel and entire MIDI.

Up to this point, we have formally defined the dissimilarity, \( d_i \), of a channel, \( c_i \), and defined distance set \( DS \). Top Rank approach is based on selecting multiple channels depending on their minimum \( d_i \) values. Since the \( DS \) is somehow a ranked list of channels showing how a channel is close to the whole, we call it as Top Rank selection. In Table 2, we present the performance of Top Rank algorithm, where top three ranked channels will be selected as melody.

Recall and precision performances of Top Rank algorithm denotes that optimal melody channels have tendency to expose shorter distances. In other words, histogram similarity between a channel and entire set is a clue for melodic content.

It is possible to improve the performance of Top Rank. Because Top Rank approach misses the melody channels which expose long distance scores. In order to catch all melody channels, clustering based approach should be useful.

Our second approach, clustering, aims to choose peculiar channels. It clusters the channels, based on their distances. Therefore, each cluster maintains peculiar histogram and distance metrics. Later, approach selects the best channel from each cluster to conserve fundamental pitch and distance features. Clustering approach is able to select the melody channels which expose long distances; at the same time, it conserves the melody channels whose distance score is shorter. To do so, it implements an agglomerative clustering approach.

Recall that Agglomerative clustering is a bottom up clustering method, which merges two nearest neighbor clusters at each step. Unless a threshold is defined, all channels will be collected in one cluster in the final. Hence, approach computes the clustering threshold by the following formulation:

Let \( \overline{h}_w \) is the weighted average pitch histogram of \( M \) such that;
\[
\overline{h}_w = \{\overline{h}_{wi1}, \overline{h}_{wi2}, ..., \overline{h}_{wi128}\} \quad \text{Where} \quad (12)
\]
\[
\overline{h}_{wi} = \sum_{k=1}^{16} h_{ki} f_i \quad (13)
\]
Threshold, \( t \), is computed as the distance between average pitch histogram, \( \overline{h}_r \), and weighted average pitch histogram, \( \overline{h}_w \). Formally;
\[
t = \frac{d(\overline{h}_w, \overline{h}_r)}{2} \quad (14)
\]
The agglomerative clustering merges clusters until the distance between two most similar clusters are less than threshold. As a result, each cluster collects the channels those have similar distances and histogram features.

After clustering the channels, we find the best channel of each cluster. For this purpose, channel selection algorithms from literature can be used. However, we introduce a new channel selection approach, Combined Selection, which produces a better result. Given that, \( a_i \) and \( b_i \) are the pitch average and predictive entropy of \( c_i \), respectively. Combined Selection computes the criterion, \( x_i \), as:
\[
x_i = a_i + b_i * 128 \quad (15)
\]
Consequently, the channel with maximum \( x_i \) is selected as melody. In Formula 15, we balance the weights of average pitch and predictive entropy. Because, pitch average, \( a_i \), ranges between 1 and 128, while predictive entropy, \( b_i \), ranges between 0 and 1.

We clarify our clustering approach with an example in Table-1. The First column of table shows the channel IDs where channels are sorted by distance column, \( d_i \). The second column shows whether the channel contains melodic information, which is set by musical experts manually. The delta column is the difference between current and previous channel distance in order. If there is a long jump between consecutive rows, it should be interpreted as the boundary of a new cluster. Hence, channels in this example should be decomposed into three basic clusters. The Cluster ID column of Table-1 shows the
clusters generated by clustering approach. The $x_i$ values of all channels are computed by Combine Selection approach and finally $c_1$, $c_3$, and $c_5$ have been chosen as representative since they have maximum $x_i$ values within respective clusters. Fragment notes of those three channels can be seen in Figure 3.

Finally, in our example, our approach keeps important melodic contents, although existing algorithms eliminates them. In summary, clustering approach can be presented in 6 basic steps:

1. Apply Skyline algorithm to all Channels
2. $\forall c_i \in M$ compute $a_i$, $b_i$ and $x_i$
3. Represent music notes with 12 semitones, compute pitch histogram set $H$.
4. Implement Agglomerative Clustering based on histogram distance.
5. Select melody channels and eliminate remaining ones.
6. Apply Skyline Algorithm.

**Table 1:** Histogram related distance features of "Always" from Bon Jovi.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Contain Melody</th>
<th>Distance ($d_i$)</th>
<th>Delta</th>
<th>Cluster ID</th>
<th>$x_i$</th>
<th>Selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>Y</td>
<td>0.0885</td>
<td>--</td>
<td>1</td>
<td>154.8</td>
<td>Y</td>
</tr>
<tr>
<td>$C_6$</td>
<td>N</td>
<td>0.0935</td>
<td>0.005</td>
<td>1</td>
<td>100.1</td>
<td>N</td>
</tr>
<tr>
<td>$C_3$</td>
<td>N</td>
<td>0.0974</td>
<td>0.003</td>
<td>1</td>
<td>107.7</td>
<td>N</td>
</tr>
<tr>
<td>$C_5$</td>
<td>N</td>
<td>0.1065</td>
<td>0.009</td>
<td>1</td>
<td>98.6</td>
<td>N</td>
</tr>
<tr>
<td>$C_4$</td>
<td>N</td>
<td>0.1092</td>
<td>0.002</td>
<td>1</td>
<td>94.0</td>
<td>N</td>
</tr>
<tr>
<td>$C_9$</td>
<td>Y</td>
<td>0.1609</td>
<td>0.051</td>
<td>2</td>
<td>102.8</td>
<td>Y</td>
</tr>
<tr>
<td>$C_5$</td>
<td>Y</td>
<td>0.2070</td>
<td>0.046</td>
<td>3</td>
<td>78.6</td>
<td>Y</td>
</tr>
<tr>
<td>$C_2$</td>
<td>N</td>
<td>0.2149</td>
<td>0.007</td>
<td>3</td>
<td>49.5</td>
<td>N</td>
</tr>
<tr>
<td>$c_{11}$</td>
<td>N</td>
<td>0.2174</td>
<td>0.002</td>
<td>3</td>
<td>74.9</td>
<td>N</td>
</tr>
</tbody>
</table>

4. Evaluation and Experimentation

In order to evaluate melody extraction algorithms, we prepared a MIDI test bed which covers fundamental music styles. Selected music styles are melody in high frequency, accompany in high frequency, melody change instrument, music in high volume, rest, arpeggio, tremolo and glissando. Our MIDI test bed consists of 31 music files. All files are selected ourselves. The entire test bed and their properties can be seen in Appendix.

It is necessary to determine optimal melody channels and their weights manually. So that performance of the channel selection algorithms can be compared with optimal selections. Moreover, selecting the melody channel which has maximum contribution is preferable in retrieval.

Our Melody Extraction application has been written in Matlab 7.0 under Windows XP. In addition, we have used Matlab MIDI Toolbox developed by Torvaldis et.al. [13] for handling MIDI files. Also, we have prepared a java application for visualization purposes as a supplementary tool.

We started our experiments with Skyline and Revised Skyline Algorithms and made analysis both visual and auditory. Then, we computed basic channel features such as total notes, pitch average value, predictive entropy, and volume and pitch histogram. Later on, we analyzed Channel Selection algorithms which come from literature. Finally, we recorded the experimental outputs of our approaches and make comparisons. Recall and precision metrics are used to evaluate channel selection algorithms. Meanwhile visual and auditory techniques are used to compare all Melody Extraction Algorithms.

There are basic pros and cons of Skyline Algorithms. Experiments indicate that Skyline Algorithm overtakes channel selection algorithms, when multiple channels contain melody. However, if accompany notes have high frequency, then both Skyline Algorithms may eliminate melody notes. Moreover, Skyline performance decreases when data set consist of music styles which are rapid succession of notes rather than concurrent notes, arpeggio and rapid succession of the same note, tremolo.

Our test results are shown in Table 2. In algorithm column, Top Channel, Entropy Channel and High Volume are existing methods. On the other hand, Combined Selection is our proposal where a single channel is selected as melody.

Our channel selection approach, Combined Selection, overtakes all previous channel selection algorithms both in terms of recall and precision. Especially precision of Combined Selection, which is 93%, is an attractive result. Because of this fact, we have used Combined Selection in clustering approach.

---

Table 2: Performance Evaluation of Channel Selection Algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Recall</th>
<th>Weighted Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best-k</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clustering</td>
<td>0.769</td>
<td>0.769</td>
<td>0.653</td>
</tr>
<tr>
<td>Top Rank</td>
<td>0.681</td>
<td>0.687</td>
<td>0.5</td>
</tr>
<tr>
<td>Combined Sel.</td>
<td>0.593</td>
<td>0.603</td>
<td>0.935</td>
</tr>
<tr>
<td>Entropy Chan</td>
<td>0.477</td>
<td>0.470</td>
<td>0.774</td>
</tr>
<tr>
<td>Top Channel</td>
<td>0.465</td>
<td>0.487</td>
<td>0.741</td>
</tr>
<tr>
<td>High Volume Channel</td>
<td>0.377</td>
<td>0.374</td>
<td>0.516</td>
</tr>
</tbody>
</table>

In Table 2, there are two different Best-k approaches. These are Clustering and Top Rank. Table 2 denotes that performance of Top Rank approach is intermediate. However, approach shows that histogram and corresponding distance features of a MIDI channels can indicate locations, where melody channels are densely populated. Consequently, Top Rank inspires Clustering approach.

In terms of recall, Clustering approach outperforms. It includes the channels which are also selected by Combined Selection. Moreover, depending on the clusters, it may select multiple melody channels.

Appendix implies that Clustering rarely misses the weighted melody channels. As a result, selected channels improve the performance of Skyline Algorithms. Moreover, reduced channel set not only prevents mess on Skyline output and preserve silent melodic intervals, but also speeds up computation time. In Figure 2, we present a fragment from example song. Although music consists of many channels/instruments, melodic information is stored in three channels. Therefore keeping notes of those three channels suffice and cost of retrieval operations will be reduced.

In terms of precision, naturally, channel selection algorithms outperform. Our Combined Selection approach is the leader on this field. Nevertheless, selecting multiple channels increases the risk of misselection. We think, Clustering approach proposes an acceptable trade off between recall and precision.

Property or style of the music has a decent factor to determine the performance of melody extraction algorithm. If tested music files have common feature such that melody has high frequency, Top Channel algorithm and Skyline Algorithm outperforms. On the contrary, if accompany has high frequency in test bed, entropy or volume based channel selection techniques will brighten. As a result, combination of channel selection algorithms outperforms when test bed contain different types of music.

Figure 2: Fragment of the original Performance of the song “Always”

Our Clustering approach considers multiple properties of music. These properties are pitch histogram, entropy and pitch average. Moreover, final channel will be exposed to Skyline Algorithm. Therefore, performance of this approach is competitive in each type of data sets.

Interestingly, Combined Selection algorithm is very successful when MIDI file exposes tremolo or arpeggio; so do clustering approach. Especially tremolo yield unique histogram structure and prevents the risk of two melody channels assigned in the same cluster. Also tests on glissando examples were successful. However, glissando alone can be a signature to contain melody.

5. CONCLUSIONS AND FUTURE WORK

Polyphonic nature of the music is the main difficulty of the music retrieval. Concurrency of notes increases the search time dramatically. One solution for the polyphonic music retrieval is melody extraction, which generates monophonic equivalent of polyphonic music files.
In this study, we presented a new Melody Extraction approach, which distinguishes melody channels of a MIDI file and eliminates the rest. In contrast to single channel selection algorithms, our approach selects Best-$k$ melody channels. Depending on predictive entropy, average pitch frequency and pitch histogram of each channel, our approach determines melody channels. Based on recall and precision metrics, we have evaluated melody extraction algorithms on the test bed. Experiments show that high precision can be achieved by combining entropy channel and top channel algorithms. Meanwhile, high recall may require selection of multiple relevant channels. To do this, additionally, we considered pitch histogram of MIDI channels. Consequently, considering multiple features improves performance.

Still, Melody extraction is an open research area. Considering only melody channels of music can speed up MIR. Such improvement requires channel selection algorithms which is able to expose high recall and precision values.

6. References


<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Description</th>
<th>k</th>
<th>Total Notes</th>
<th>Optimal Chan.</th>
<th>Weights</th>
<th>Best-k</th>
<th>Combined Selection</th>
<th>Top Channel</th>
<th>Entropy Channel</th>
<th>High Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>1G1</td>
<td>Mozart Alla Turka</td>
<td>Melody has high frequency</td>
<td>2</td>
<td>711</td>
<td>1</td>
<td>100</td>
<td>1,2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1G2</td>
<td>Chopin - Etud Op. 12</td>
<td>*</td>
<td>1</td>
<td>471</td>
<td>1</td>
<td>100</td>
<td>1,2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1G3</td>
<td>Mozart -12 Variations</td>
<td>*</td>
<td>2</td>
<td>134</td>
<td>1</td>
<td>100</td>
<td>1,2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2G1</td>
<td>Beethoven 5. Symph</td>
<td>Moodi change instrument</td>
<td>1</td>
<td>1011</td>
<td>83,5,1</td>
<td>40-30-20-10</td>
<td>4,11,3</td>
<td>4,8,12</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>2G2</td>
<td>Mozart -12 Variations</td>
<td>*</td>
<td>2</td>
<td>318</td>
<td>1,2</td>
<td>60-40</td>
<td>1,2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2G3</td>
<td>Mozart-Concerto</td>
<td>*</td>
<td>9</td>
<td>428</td>
<td>12,6,9</td>
<td>40-30-20-10</td>
<td>1,3,2</td>
<td>1,3,2,8</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3G1</td>
<td>Madonna F Roxette</td>
<td>Accompany has high frequency</td>
<td>1</td>
<td>2357</td>
<td>4</td>
<td>100</td>
<td>4,12,2</td>
<td>12,7,5,9</td>
<td>12</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>3G2</td>
<td>Roxette- It must be love</td>
<td>*</td>
<td>8</td>
<td>2235</td>
<td>2</td>
<td>60-40</td>
<td>8,3,5,1</td>
<td>8,7,3</td>
<td>6,5,9,16</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>4G1</td>
<td>Bach - Goldberg</td>
<td>Rest</td>
<td>1</td>
<td>283</td>
<td>1</td>
<td>100</td>
<td>1,2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4G2</td>
<td>R. Martin - Maria</td>
<td>*</td>
<td>8</td>
<td>2474</td>
<td>16,3</td>
<td>60-40</td>
<td>6,7,3</td>
<td>6,5,9,16</td>
<td>6</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>4G3</td>
<td>Yanni - Themes</td>
<td>*</td>
<td>4</td>
<td>670</td>
<td>1,12</td>
<td>60-40</td>
<td>12,1,2</td>
<td>12,1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5G1</td>
<td>Green Sleeves</td>
<td>Appeggio</td>
<td>1</td>
<td>183</td>
<td>1</td>
<td>100</td>
<td>1,2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5G2</td>
<td>Rodgers - Romani</td>
<td>*</td>
<td>3</td>
<td>338</td>
<td>1</td>
<td>100</td>
<td>1,2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5G3</td>
<td>Albeniz - Asturias</td>
<td>*</td>
<td>2</td>
<td>463</td>
<td>2</td>
<td>100</td>
<td>1,2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>6G1</td>
<td>Pulp Fiction</td>
<td>Tremolo</td>
<td>4</td>
<td>699</td>
<td>1,6,13,4</td>
<td>50-30-20-10</td>
<td>5,6,3</td>
<td>5,6</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>6G2</td>
<td>Vivaldi - Concerto</td>
<td>*</td>
<td>6</td>
<td>412</td>
<td>1</td>
<td>100</td>
<td>1,2</td>
<td>1,6,3</td>
<td>1,6,3,2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6G3</td>
<td>Empire Strikes Back</td>
<td>*</td>
<td>1</td>
<td>532</td>
<td>39,4</td>
<td>50-30-20</td>
<td>3,4,1</td>
<td>3,9,8</td>
<td>9</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>7G1</td>
<td>Tchaikovski - 1812</td>
<td>Volume</td>
<td>1</td>
<td>2119</td>
<td>1,6,3</td>
<td>50-30-20-10</td>
<td>1,6,3</td>
<td>1,7,16,4</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>7G2</td>
<td>F. Lizst - Totentanz</td>
<td>*</td>
<td>4</td>
<td>1480</td>
<td>1,4</td>
<td>60-40</td>
<td>2,3,1</td>
<td>2,4</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>7G3</td>
<td>M. Jackson - Thriller</td>
<td>*</td>
<td>7</td>
<td>989</td>
<td>4</td>
<td>100</td>
<td>3,4,1</td>
<td>4,5,7</td>
<td>4</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>8G1</td>
<td>C. Dion - All By Myself</td>
<td>Glissando</td>
<td>5</td>
<td>212</td>
<td>7,8,1</td>
<td>50-30-20</td>
<td>1,7,8</td>
<td>1,7,2</td>
<td>7</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>8G2</td>
<td>Eric Clapton-Tears in Conga</td>
<td>*</td>
<td>5</td>
<td>241</td>
<td>5</td>
<td>100</td>
<td>5,7,2</td>
<td>5,7,2</td>
<td>5</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>8G3</td>
<td>C. Dion - All By Myself</td>
<td>Glissando</td>
<td>3</td>
<td>4506</td>
<td>21,5</td>
<td>50-30-20</td>
<td>3,1,2</td>
<td>3,1</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>MG1</td>
<td>Mozart - 40 Symph.</td>
<td>Mixed</td>
<td>1</td>
<td>1365</td>
<td>1,2,3,4,1</td>
<td>40-30-20-5</td>
<td>13,15,16</td>
<td>13,1</td>
<td>12</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>MG2</td>
<td>Tears in heaven</td>
<td>*</td>
<td>8</td>
<td>9290</td>
<td>4,5</td>
<td>60-40</td>
<td>7,6,5</td>
<td>6,5,4,8</td>
<td>5</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>MG3</td>
<td>Celine Dion- All By One</td>
<td>*</td>
<td>9</td>
<td>3003</td>
<td>4,7,8,1</td>
<td>40-30-20-10</td>
<td>5,7,8</td>
<td>8,1,6</td>
<td>1</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>MG4</td>
<td>Beethoven - Pastoral</td>
<td>*</td>
<td>1</td>
<td>4721</td>
<td>11,2,3,4</td>
<td>40-30-20-5</td>
<td>2,1,4</td>
<td>1,3,11,6</td>
<td>11</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>MG5</td>
<td>Bye bye love</td>
<td>*</td>
<td>6</td>
<td>4369</td>
<td>4</td>
<td>100</td>
<td>3,2,5</td>
<td>3,2,4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>MG6</td>
<td>Celine Dion - All By One</td>
<td>*</td>
<td>1</td>
<td>6025</td>
<td>13,6,1,3</td>
<td>40-30-20-10</td>
<td>6,1,8</td>
<td>1,8,13,7</td>
<td>1</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>MG7</td>
<td>Bon Jovi - Always</td>
<td>*</td>
<td>1</td>
<td>6527</td>
<td>8,5,1</td>
<td>50-30-20</td>
<td>1,6,3</td>
<td>1,8,5</td>
<td>1</td>
<td>8</td>
<td>1</td>
</tr>
</tbody>
</table>

**Appendix**: Detailed Description of Test-bed used for evaluation. First column is the file id. Third column specifies the music style. Fourth column denotes total non empty channels in the MIDI. Sixth and seventh channels points out optimal melody channels and their weights respectively. Remaining columns denote the selected channels of specified algorithm.