Stream Segregation Algorithm for Pattern Matching in Polyphonic Music Databases

Wai Man Szeto and Man Hon Wong
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{wmszeto, mhwong}@cse.cuhk.edu.hk
Keywords: Multimedia databases, music information retrieval, clustering algorithms

Abstract

As music can be represented symbolically, most of the existing methods extend some string matching algorithms to retrieve musical patterns in a music database. However, not all retrieved patterns are perceptually significant because some of them are, in fact, inaudible. Music is perceived in groupings of musical notes called *streams*. The process of grouping musical notes into streams is called *stream segregation*. Stream-crossing musical patterns are perceptually insignificant and should be pruned from the retrieval results. This can be done if all musical notes in a music database are segregated into streams and musical patterns are retrieved from the streams. Findings in auditory psychology are utilized in this paper, in which stream segregation is modelled as a clustering process and an adapted single-link clustering algorithm is proposed. Supported by experiments on real music data, streams are identified by the proposed algorithm with considerable accuracy.

1 Introduction

As music can be represented in a series of musical notes, most of the existing methods extend some string matching algorithms to retrieve musical patterns

in a music database [6, 4, 5, 11, 26, 12, 10, 21, 18, 17], in which musical pitches (certain frequencies) are encoded in a finite alphabet set Σ and each alphabet is represented by an integer.

Music databases can be classified according to the types of music stored: monophonic and polyphonic. If only one pitch is played at a time throughout the music, the music is said to be *monophonic*. If the duration of each musical note is omitted and only the sounding order of pitches is retained, monophonic music can be represented as a pitch sequence $P = \langle p_1, p_2, \ldots, p_m \rangle$ where p_i is a pitch in the alphabet set Σ. If several pitches may be played simultaneously, the music is said to be *polyphonic*. The simultaneous pitches are put into a pitch set. Polyphonic music can be represented as a sequence of pitch sets, denoted by $\mathcal{S} = \langle S_1, S_2, \ldots, S_n \rangle$ where $S_i = \{s_{i,1}, s_{i,2}, \ldots, s_{i,l_i}\}$ and $s_{i,j} \in \Sigma$. In other words, in the first time interval, the pitches in $\mathcal{S}_1$ are played. In the next time interval, the pitches in $\mathcal{S}_2$ are played and so on. In fact, according to this definition, most music is polyphonic.

In general, recent contributions on pattern matching in polyphonic music databases mostly focus on the problem below: Given a monophonic query sequence $Q = \langle q_1, q_2, \ldots, q_m \rangle$ and a polyphonic source sequence $\mathcal{S} = \langle S_1, S_2, \ldots, S_n \rangle$, the aim is to search for source subsequences $\langle S_i, S_{i+1}, \ldots, S_{i+m-1} \rangle$ that matches Q such that $q_k \in S_{i+k-1}$ for $k = 1, 2, \ldots, m$. In addition to this exact matching of a query sequence, some researchers also consider vertical shifting (transposition) and edit distance [26, 11, 12, 10].

However, not all matched patterns are perceptually significant, i.e. the patterns are inaudible. In particular, the repetitive musical pattern discovery algorithm in [19] discovers over 70000 patterns in Sergei Rachmaninoff's Prelude in C sharp minor, Op. 3 No. 2 (about-4-minute piano piece). However, the authors report that probably less than 100 of these are perceptually significant.

Perceptually insignificant patterns are retrieved because music perception has not receive the attention it deserves in many existing retrieval systems. When listening to music, people perceive music in groupings of musical notes called *streams* which are the perceptual impression of connected series of musical notes, instead of isolated sounds. The process of grouping musical notes into streams is called *stream segregation*[1]. For example, a melody is usually perceived as a single coherent and continuous musical line, that is, a stream. Monophonic musical patterns across streams are perceptually insignificant and should be pruned from searching results. Consider the opening theme of

[1]In [2], *stream segregation* is divided into two processes: simultaneous integration and sequential integration. Simultaneous integration is the process of grouping frequencies into musical notes. Sequential integration is the process of grouping musical notes into streams. However, in this paper, *stream segregation* is referred to the latter process only.

Ludwig van Beethoven's Symphony No. 5, i.e. G, G, G, E♭, F, F, F, and D (or mi, mi, mi, doh, ray, ray, ray, and te) as shown in Figure 1, if we search for any of its transposition, a matched pattern is found in the bars 1 to 2 of Frederic Chopin's Prelude No. 4 in E minor (Figure 2). However, this pattern is hardly audible because as shown in Figure 2, there are four streams or musical lines in the opening [23]. We perceive stream 1 as the melody, while the other three streams are combined and formed a single accompaniment. The pattern is across streams 3 and 4 so it is perceptually insignificant[2].

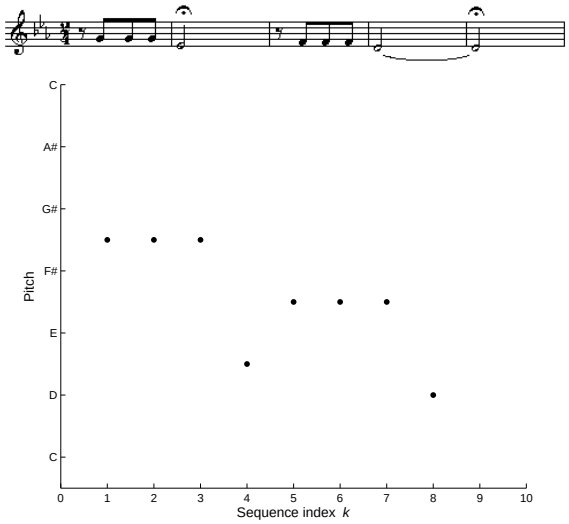


Figure 1: The opening of Beethoven's Symphony No. 5, first movement and its monophonic sequence representation $Q = \langle q_k \rangle$ for $k = 1, \ldots, 8$.

Pruning stream-crossing monophonic patterns can be done if all streams in a music database are segregated and musical patterns are retrieved from the streams. Therefore, stream segregation should be added as a pre-processing step in existing retrieval systems in order to improve the quality of retrieval. Findings in auditory psychology are utilized in this paper, in which stream segregation is modelled as a clustering process and an adapted single-link clustering algorithm is proposed. The rest of the paper is organized as follows. Section 2 provides the psychological perspective of stream segregation. Related work is reviewed in Section 3. We will present our method in Section 4. The application of stream segregation to polyphonic music databases

[2]The musical excerpts can be heard at the demonstration web site http://www.cse.cuhk.edu.hk/~wmszeto/stream/index.html.

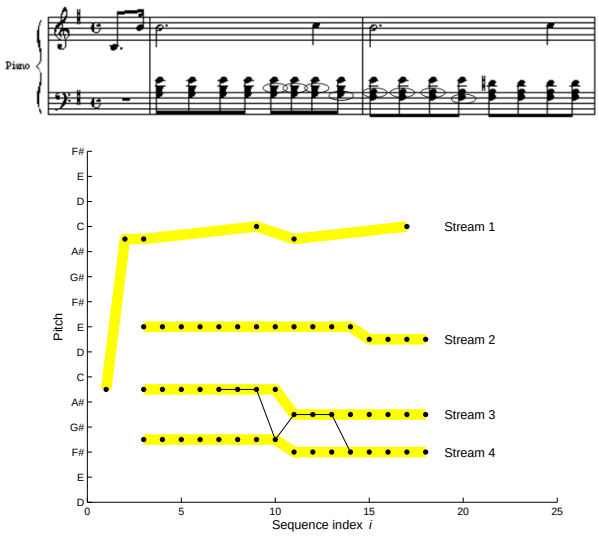


Figure 2: The opening of Chopin's Prelude No. 4 in E minor and its monophonic sequence representation $\mathcal{S} = \langle S_i \rangle$ for $i = 1, \ldots, 18$. The circles in the staff and the solid line in the sequence representation correspond to the Beethoven's symphony.

is discussed in Section 5. In Section 6, we show our experimental results. Finally, a conclusion is given in Section 7.

2 Findings in auditory psychology

Extensive psychological experiments have been being designed to study the factors of stream segregation. The following findings are summarized from the surveys and the findings in [24, 2, 27, 7]. Most studies of stream segregation in psychology are influenced by the "Gestalt principles of perception" proposed in the 1920s. It is a set of principles governing the grouping of elements in perception. One of the principles is proximity: closer elements are grouped together in preference to those that are spaced further apart.

It has been found that the pitch proximity and the temporal proximity are dominant factors in stream segregation. Pitch proximity states that musical notes which are closer in terms of pitch are grouped together. When a sequence of two alternating pitches A and B is played as $ABAB\ldots$, the two pitches will seem to fuse into a single stream $ABAB\ldots$ if they are close together in pitch (Figure 3(a)); otherwise they will seem to form two independent streams $AA\ldots$ and $BB\ldots$ (Figure 3(b)). For temporal proximity,

it states that musical notes which are closer in terms of time are grouped together. When a sequence of two alternating pitches is played, we tend to perceive them as a single stream if the alternation speed is low (Figure 4(a)). If the alternation speed is high, the two pitches will form two independent streams (Figure 4(b)). Although timbre, amplitude, and spatial location may sometimes affect stream segregation, they are often outweighed by pitch and temporal proximities.

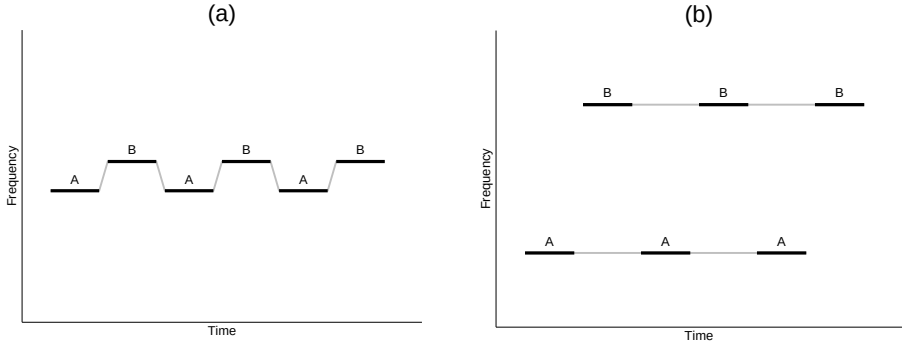


Figure 3: Illustrations of the pitch proximity (adapted from [2, 27]).

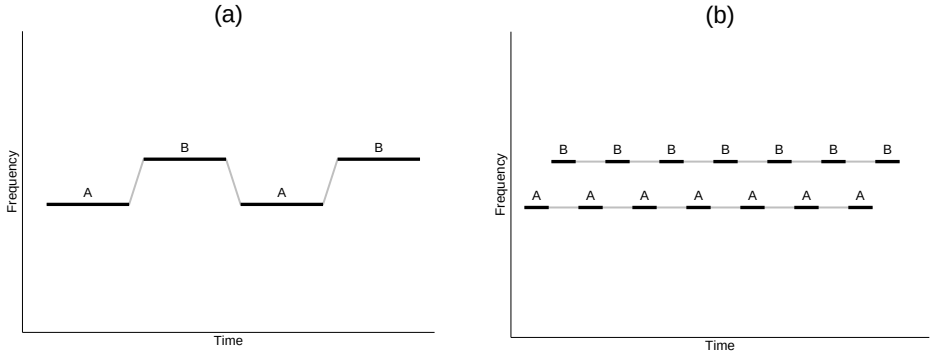


Figure 4: Illustrations of the temporal proximity (adapted from [2, 27]).

In fact, there are three basic assumptions in many psychological experiments of stream segregation [24]. The first assumption is that listeners tend to assign a musical note to only one stream. A musical note would not belong to more than one stream. The second assumption is that for a given stream, there will not be more than one musical note at the same time. A stream would not contain any simultaneous musical notes. Hence, streams are monophonic. The third assumption states that there is a preference to perceive fewer streams. We tend to minimize the number of streams.

3 Related work

Various methods have been proposed for stream segregation and some of which are reviewed in [24]. Based on an analogy with the apparent motion in vision, Gjerdingen proposes a massively parallel, multiplex, feed-forward neural-network system in [8]. In the system, the input is musical notes which are discrete events represented in a pitch-time plane. The "influence" of an event is measured as activation. An event has greatest activation at its pitch value and during its time span. Its influence also "diffuses" to surroundings. A temporal filter smooths the activations of all events in the time dimension. A spatial filter diffuses events over a wider span of pitches with the assumption of a Gaussian distribution of the diffusion. The hill-like activation of all events are summed up. Tracking local maxima of activation along the time dimension can produce a pitch-time graph containing lines, which are interpreted as streams. The model can also track more than one line simultaneously. However, the lines in the pitch-time graph may be difficult to interpret.

The McCabe and Denham's model in [16] is similar to Gjerdingen's. Like Gjerdingen's model, the McCabe-Denham model represents music in the dimensions of pitch and time and uses neural networks to find out streams. There are two neural networks. One neural network corresponds to the "foreground" stream and another corresponds to the "background" stream. Musical input is segregated into the two streams. Although the McCabe-Denham model accepts acoustic input, complex acoustic signals have not been tested. Similar to Gjerdingen's model, the graphical output of McCabe-Denham model may be difficult to interpret. Moreover, it can track only two streams.

The latest model, developed by Temperley [24], is based on a set of stream segregation preference rules adapted from Lerdahl and Jackendoff [13], and is built on a rule-based stream segregation model in [14]. Its input is discrete pitch events. Pitches are in the unit of MIDI pitch number, which is one of the integers ranging from 0 to 127 inclusively. Middle C is assigned with the value 60. The time dimension is quantized. Hence, music is represented in a pitch-time grid of squares. If there is a note, the square is "black". If not, the square is "white". The model has six mandatory preference rules and one optional preference rule. A user is required to input the weight of each rule and the maximum number of streams. Then the system searches for the optimal stream configuration using dynamic programming. However, overlapping notes of the same pitch are disallowed in the input representation and there is no known experimental evidence to show that MIDI pitch number is an appropriate perceptual distance.

4 Proposed method

According to the findings in auditory psychology discussed in Section 2, there are four important preferences in stream segregation: (i) the pitch proximity and the temporal proximity are dominant factors; (ii) a musical note is assigned to only one stream but not more; (iii) a stream does not contain any simultaneous musical notes; and (iv) the number of streams is minimized. Based on these findings, we represent music as events and propose a clustering model for stream segregation.

4.1 Music representation

As pitch and temporal proximities are dominant, the attributes of pitch and time of music are taken into account. Each musical note is represented as an *event*. An event $\mathbf{e}$ is a vector (t^s, t^e, p), where t^s is the start time, t^e is the end time, and p is the pitch. The start time and the end time are in the unit of seconds. The pitch is represented as the frequency in the *mel scale* [20, 3]. The mel scale represents the average judgement of the relation between perceptual pitch and frequency, where the scale is adjusted such that 1000 mels map to 1 kHz. The mapping is approximately linear in frequency up to 1 kHz, and logarithmic beyond 1 kHz. The scale is defined in the following analytical expression:

$$f_{mel} = 2595 \cdot \log_{10}(1 + \frac{f_{Hz}}{700})$$

An example is shown in Figure 5 and Table 1.

$\mathbf{e}_i$	Pitch name	(t^s, t^e, p)
$\mathbf{e}_1$	C4	$(0, 2, 357.87)$
$\mathbf{e}_2$	G4	$(0, 0.5, 501.15)$
$\mathbf{e}_3$	A♭4	$(1, 3, 524.96)$
$\mathbf{e}_4$	E4	$(2, 4, 434.88)$
$\mathbf{e}_5$	F4	$(3, 4, 456.13)$

Table 1: The events in the five-event example.

4.2 Algorithm

Intuitively, if musical notes are represented as events in the pitch-time space, the "distance" between two events can reflect the pitch proximity and the temporal proximity of the corresponding musical notes. A stream, a group of

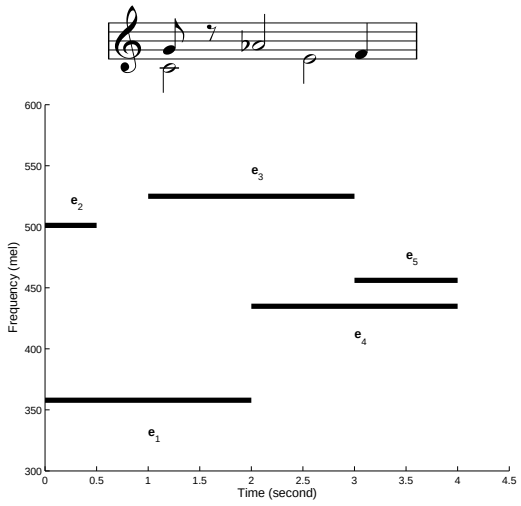


Figure 5: A five-event example. The speed is 60 crotchet beats per minute.

events sharing "similar" pitch attribute and time attribute, is in fact a cluster, so stream segregation can be modelled as a clustering problem. A stream is thus referred to as a cluster hereinbelow. However, traditional clustering techniques cannot be directly applied to stream segregation because they handle data points rather than events. Moreover, besides the pitch and the temporal proximities, the findings in auditory psychology show that there are preferences to include each event in only a single stream and to avoid multiple simultaneous events in a single stream. To include these findings, we define two kinds of the relation between two events: *sequential events* and *simultaneous events*. Given two events, if their durations overlap each other, they are simultaneous events. Otherwise, they are sequential events. The definitions are formally given below:

Definition 4.1 (Sequential events) *Given two events* $\mathbf{e}_1 = (t_1^s, t_1^e, p_1)$ *and* $\mathbf{e}_2 = (t_1^s, t_2^e, p_2)$*, they are* sequential events *if* $t_1^e \leq t_2^s$ *or* $t_2^e \leq t_1^s$*, denoted by* $\mathbf{e}_1 \nparallel \mathbf{e_2}$*. If* $t_1^e \leq t_2^s$*, then* $\mathbf{e}_1 \prec \mathbf{e}_2$*. If* $t_2^e \leq t_1^s$*, then* $\mathbf{e}_1 \succ \mathbf{e}_2$*.*

Definition 4.2 (Simultaneous events) *If two events* $\mathbf{e}_1$ *and* $\mathbf{e}_2$ *are not sequential events, they are said to be* simultaneous events*, denoted by* $\mathbf{e}_1 \parallel \mathbf{e}_2$*.*

These definitions are further illustrated by the following example:

Example 4.1 *In Figure 5, the event pairs* $\{\mathbf{e}_1, \mathbf{e}_4\}$*,* $\{\mathbf{e}_1, \mathbf{e}_5\}$*,* $\{\mathbf{e}_2, \mathbf{e}_3\}$*,* $\{\mathbf{e}_2, \mathbf{e}_4\}$*,* $\{\mathbf{e}_2, \mathbf{e}_5\}$*, and* $\{\mathbf{e}_3, \mathbf{e}_5\}$ *are sequential events because they do not overlap each other in time. The event pairs* $\{\mathbf{e}_1, \mathbf{e}_2\}$*,* $\{\mathbf{e}_1, \mathbf{e}_3\}$*,* $\{\mathbf{e}_3, \mathbf{e}_4\}$*, and* $\{\mathbf{e}_4, \mathbf{e}_5\}$ *are simultaneous events.*

After defining the relations between events, we define the distance function between two events. To avoid putting simultaneous events in the same cluster, the distance between any two simultaneous events is set to infinity. Hence, a cluster cannot contain any simultaneous event. For sequential events, it is the distance between the tail of the prior event and the head of the posterior event. In order to define the concept formally, we need to address two problems. The first one is the choice of metrics. However, a definitive answer to the most appropriate choice of metrics to "musical" space, to our knowledge, has never been done. Therefore, we use the Euclidean distance, the most common metric. The second problem is the difference of measurement units of pitch and time. We tackle it by assigning weighting factors to the time dimension and the pitch dimension. The weighting factors will be determined empirically. The inter-event distance function is defined below:

Definition 4.3 (Inter-event distance) *Given two events* $\mathbf{e}_1 = (t_1^s,\ t_1^e, p_1)$ *and* $\mathbf{e}_2 = (t_1^s, t_2^e, p_2)$*, the* inter-event distance $EDIST$ *is*

$$EDIST(e_1, e_2) = \begin{cases} \sqrt{(\alpha d)^2 + (\beta(p_1 - p_2))^2} & \text{if } \mathbf{e}_1 \nparallel \mathbf{e_2}, \\ \infty & \text{if } \mathbf{e}_1 \parallel \mathbf{e_2}. \end{cases}$$

where α *is the time weighting factor,* β *is the pitch weighting factor, and*

$$d = \begin{cases} t_1^e - t_2^s & \text{if } \mathbf{e}_1 \prec \mathbf{e}_2, \\ t_2^e - t_1^s & \text{if } \mathbf{e}_1 \succ \mathbf{e}_2. \end{cases}$$

Example 4.2 *Suppose* $\alpha = 1$ *and* $\beta = 1$*. The inter-event distances between all possible pairs of the events in Figure 5 are shown in Table 2* [3]*.*

	$\mathbf{e}_1$	$\mathbf{e}_2$	$\mathbf{e}_3$	$\mathbf{e}_4$	$\mathbf{e}_5$
$\mathbf{e}_1$	−	∞	∞	77.11	98.39
$\mathbf{e}_2$	∞	−	23.83	66.36	45.14
$\mathbf{e}_3$	∞	23.83	−	∞	68.89
$\mathbf{e}_4$	77.11	66.36	∞	−	∞
$\mathbf{e}_5$	98.39	45.14	68.89	∞	−

Table 2: The inter-event distances of the events in Figure 5 ($\alpha = 1$; $\beta = 1$).

Before going further, we define some notations adapted from those of the traditional clustering techniques in [25]. A piece of music E consists of N

[3]The self-distance of events is not shown and is marked in "−" because an event does not form a cluster with itself.

events:

$$E = \{\mathbf{e}_i \mid i = 1, 2, \ldots, N\}$$

A cluster $\mathcal{C}$ has the start time T^s, the end time T^e, and a set of clustered events C:

$$\mathcal{C} = (T^s, T^e, C)$$

where

$$\begin{aligned} C &= \{\mathbf{e}_{i_1}, \mathbf{e}_{i_2}, \ldots\} \subseteq E \\ &\qquad \text{where } \mathbf{e}_{i_k} = (t^s_{i_k}, t^e_{i_k}, p_{i_k}), \\ T^s &= \min_{i_k}(t^s_{i_k}), \\ T^e &= \max_{i_k}(t^e_{i_k}). \end{aligned}$$

A clustering $\mathfrak{R}$ contains n clusters:

$$\mathfrak{R} = \{\mathcal{C}_j \mid j = 1, 2, \ldots, n\}$$

As streams have chain-like shapes, we adapt the single-link clustering algorithm for solving the stream segregation problem. An event is regarded as a data point. The single-link clustering algorithm is a hierarchical clustering method, which generates a nested series of partitions of data points. In the single-link method, the distance between two clusters is the minimum of the distances among all pairs of data points drawn from the two clusters (one data point from the first cluster, the other from the second) [9]. In order to avoid a cluster containing any simultaneous events, we define the following two relations between two clusters similar to those between two events.

Definition 4.4 (Sequential clusters) *Given two clusters* $\mathcal{C}_1 = (T^s_1, T^e_1, C_1)$ *and* $\mathcal{C}_2 = (T^s_2, T^e_2, C_2)$*, they are* sequential clusters *if* $T^e_1 \leq T^s_2$ *or* $T^e_2 \leq T^s_1$*, denoted by* $\mathcal{C}_1 \nparallel \mathcal{C}_2$*.*

Definition 4.5 (Simultaneous clusters) *If two clusters* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *are not sequential clusters, they are said to be* simultaneous clusters*, denoted by* $\mathcal{C}_1 \parallel \mathcal{C}_2$*.*

The inter-cluster distance between simultaneous clusters should be set to infinity to ensure that no pair of simultaneous events is in the same cluster. For two sequential clusters, the distance between them is the minimum of the distances among all pairs of events drawn from one cluster and another cluster. We define the inter-cluster distance below:

Definition 4.6 (Inter-cluster distance) *Given two clusters* $\mathcal{C}_1 = (T_1^s, T_1^e, C_1)$, $\mathcal{C}_2 = (T_2^s, T_2^e, C_2)$, *the* inter-cluster distance $CDIST$ *is*

$$CDIST(\mathcal{C}_1, \mathcal{C}_2) = \begin{cases} \min_{\mathbf{e}_i \in C_1, \mathbf{e}_j \in C_2}(EDIST(\mathbf{e}_i, \mathbf{e}_j)) & \text{if } \mathcal{C}_1 \nparallel \mathcal{C}_2, \\ \infty & \text{if } \mathcal{C}_1 \parallel \mathcal{C}_2. \end{cases}$$

There are various methods to implement the single-link clustering algorithm. Here, we adapt the agglomerative algorithm in [25, 15, 9]. For the agglomerative algorithm, the initial clustering $\mathfrak{R}_0$ consists of N clusters, each of which contains one event in E. Among all possible pairs of clusters, find the pair that has the minimum inter-cluster distance over all other pairs. Then the pair of clusters is merged into a larger cluster. Thus, a new clustering $\mathfrak{R}_1$ is formed and the process is repeated. Following the psychological preference that the number of streams is minimized, we minimize the number of clusters by setting the two termination conditions as follows. The first condition is that the distances of all possible pairs of clusters are infinity. This means that all pairs of clusters are simultaneous clusters. Noted that during the clustering process, only sequential clusters but not simultaneous clusters can be merged together. Hence, the resulting clusters will not contain any simultaneous events. The second condition is that all events lie in the same cluster, i.e. one large cluster contains all events, in which any two events must be sequential events. As a result, users are not required to input the threshold or the number of clusters. The adapted single-link clustering is shown in Algorithm 1.

Algorithm 1 Adapted single-link clustering algorithm

1: Choose $\mathfrak{R}_0 \leftarrow \{\mathcal{C}_i = (t_i^s, t_i^e, \{\mathbf{e}_i\}), i = 1, 2, \ldots, N\}$ as the initial clustering.
2: $t \leftarrow 0$
3: **repeat**
4: $\quad t \leftarrow t + 1$
5: $\quad$ Among all possible pairs of clusters $(\mathcal{C}_r, \mathcal{C}_s)$ in $\mathfrak{R}_{t-1}$, find $\mathcal{C}_i$, $\mathcal{C}_j$ such that $CDIST(\mathcal{C}_i, \mathcal{C}_j) = \min_{r \neq s} CDIST(\mathcal{C}_r, \mathcal{C}_s)$
6: $\quad$ **if** $CDIST(\mathcal{C}_i, \mathcal{C}_j) = \infty$ **then**
7: $\quad\quad$ **break**
8: $\quad$ Merge $\mathcal{C}_i$, $\mathcal{C}_j$ into a single cluster $\mathcal{C}_q$ and form $\mathfrak{R}_t \leftarrow (\mathfrak{R}_{t-1} - \{\mathcal{C}_i, \mathcal{C}_j\}) \cup \mathcal{C}_q$.
9: **until** $\mathfrak{R}_{N-1}$ clustering is formed, i.e., all events lie in the same cluster.

Generated by the algorithm running on the example in Figure 5 with $\alpha = 1$ and $\beta = 1$, the iterative clustering results are shown in Table 3. The

clustering of the five-event example is presented in Figure 6, where $\{\mathbf{e}_1, \mathbf{e}_4\}$ is a cluster and $\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$ is another.

	$\{\mathbf{e}_1\}$	$\{\mathbf{e}_2\}$	$\{\mathbf{e}_3\}$	$\{\mathbf{e}_4\}$	$\{\mathbf{e}_5\}$
$\{\mathbf{e}_1\}$	–	∞	∞	77.11	98.39
$\{\mathbf{e}_2\}$	∞	–	23.83	66.36	45.14
$\{\mathbf{e}_3\}$	∞	23.83	–	∞	68.89
$\{\mathbf{e}_4\}$	77.11	66.36	∞	–	∞
$\{\mathbf{e}_5\}$	98.39	45.14	68.89	∞	–

(a) $t = 0$

	$\{\mathbf{e}_1\}$	$\{\mathbf{e}_2, \mathbf{e}_3\}$	$\{\mathbf{e}_4\}$	$\{\mathbf{e}_5\}$
$\{\mathbf{e}_1\}$	–	∞	77.11	98.39
$\{\mathbf{e}_2, \mathbf{e}_3\}$	∞	–	∞	45.14
$\{\mathbf{e}_4\}$	77.11	∞	–	∞
$\{\mathbf{e}_5\}$	98.39	45.14	∞	–

(b) $t = 1$

	$\{\mathbf{e}_1\}$	$\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$	$\{\mathbf{e}_4\}$
$\{\mathbf{e}_1\}$	–	∞	77.11
$\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$	∞	–	∞
$\{\mathbf{e}_4\}$	77.11	∞	–

(c) $t = 2$

	$\{\mathbf{e}_1, \mathbf{e}_4\}$	$\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$
$\{\mathbf{e}_1, \mathbf{e}_4\}$	–	∞
$\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$	∞	–

(d) $t = 3$

Table 3: Inter-cluster distance and clustering at different t of the five-event example.

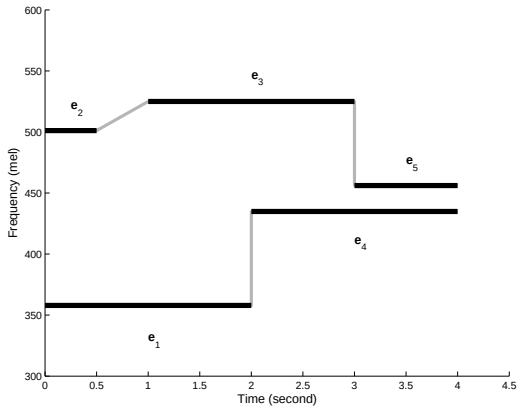


Figure 6: Clustering result: $\{\mathbf{e}_1, \mathbf{e}_4\}$ is a cluster and $\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$ is another.

Motivated by the analysis of the general single-link method in [25], the complexity of the adapted algorithm is analyzed as follows. At each level t, there are $N - t$ clusters. Thus, in order to determine the pair of clusters that is going to be merged at the $(t + 1)$th level, the number of pairs of clusters

to be considered is

$$\binom{N-t}{2} = \frac{(N-t)(N-t-1)}{2}.$$

Therefore, if the number of output clusters is n, the total number of pairs that have to be examined throughout the whole clustering process is

$$\begin{aligned}\sum_{t=0}^{N-n}\binom{N-t}{2} &= \sum_{k=n}^{N}\binom{k}{2} \\ &= \sum_{k=0}^{N}\binom{k}{2} - \sum_{k=0}^{n-1}\binom{k}{2} \\ &= \frac{(N-1)N(N+1)}{6} \\ &\quad -\frac{(n-2)(n-1)n}{6}\end{aligned}$$

As n is small comparing to N, the total number of operations is proportional to N^3. The calculation of the inter-event distance takes constant time. Therefore, the complexity of the algorithm is $O(N^3)$.

5 Application of Stream Segregation to Polyphonic Databases

Recapitulating the exact musical pattern matching problem stated in Section 1 that given a monophonic pitch query sequence $Q = \langle q_1, q_2, \ldots, q_m\rangle$ and a polyphonic pitch-set source sequence $\mathcal{S} = \langle S_1, S_2, \ldots, S_n\rangle$ where $S_i = \{s_{i,1}, s_{i,2}, \ldots, s_{i,l_i}\}$ and $q_i, s_{i,j}$ are in the finite alphabet set Σ, the aim is to search for source subsequences $\langle S_i, S_{i+1}, \ldots, S_{i+m-1}\rangle$ that matches Q such that $q_k \in S_{i+k-1}$ for $k = 1, 2, \ldots, m$. In order to avoid retrieving perceptually insignificant patterns, we propose that stream segregation is added as the pre-processing step in a retrieval system, i.e., all polyphonic sources are segregated into streams and the streams are stored in a database. Then patterns are retrieved from the streams. As streams are monophonic, stream segregation actually reduces a polyphonic music database to a monophonic one. The pitches of streams are extracted to form pitch source sequences. The ordering of pitches follows their appearance order.

Example 5.1 *In Figure 6,* $\{\mathbf{e}_1, \mathbf{e}_4\}$ *is a stream and* $\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$ *is another stream. The pitch of* $\mathbf{e}_i$ *is* p_i *where* $p_i \in \Sigma$*. As* $\mathbf{e}_1 \prec \mathbf{e}_4$ *and* $\mathbf{e}_2 \prec \mathbf{e}_3 \prec \mathbf{e}_5$*, the two streams become two pitch source sequences* $\langle p_1, p_4\rangle$ *and* $\langle p_2, p_3, p_5\rangle$*.*

Then the exact musical pattern matching problem is redefined as follows: Given a pitch query sequence $Q = \langle q_1, q_2, \ldots, q_m \rangle$ and a pitch source sequence $S = \langle s_1, s_2, \ldots, s_n \rangle$ where $q_i, s_i \in \Sigma$, the aim is to search for source subsequences $\langle s_i, s_{i+1}, \ldots, s_{i+m-1} \rangle$ that matches Q such that $q_k = s_{i+k-1}$ for $k = 1, 2, \ldots, m$. This is like a substring matching problem. Moreover, no matched pattern is across streams so the opening theme of Beethoven's Symphony No. 5 cannot be found in Chopin's Prelude No. 4 in E minor (Figure 2) if the streams of Chopin's Prelude are correctly segregated.

The correctness of stream segregation greatly affects the accuracy of the stream-segregated retrieval system. If an event is wrongly classified into a stream, matched patterns may be missed and false alarms may occur. Hit rate and false alarm rate are proposed to evaluate the quality of stream segregation. Before introducing them, we place a total ordering on events for a simpler notation in later discussions. Events are first sorted by the start time in the ascending order. If two events have the same start time, the order is determined arbitrarily. The index i of the event $\mathbf{e}_i$ reflects this total ordering: given a music $E = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N\}$, the total ordering is $\mathbf{e}_1 < \mathbf{e}_2 < \cdots < \mathbf{e}_N$. For example, the index of each event in the five-event example in Figure 5 has been assigned according to the total ordering such that $\mathbf{e}_1 < \mathbf{e}_2 < \mathbf{e}_3 < \mathbf{e}_4 < \mathbf{e}_5$.

The correctness of stream segregation is measured in terms of *linkage* between two events. Given a stream $C = \{\mathbf{e}_{i_1}, \mathbf{e}_{i_2}, \ldots, \mathbf{e}_{i_k}\}$ where $i_1 < i_2 < \cdots < i_k$, there exists linkages in the pairs $(\mathbf{e}_{i_1}, \mathbf{e}_{i_2}), (\mathbf{e}_{i_2}, \mathbf{e}_{i_3}), \ldots,$ $(\mathbf{e}_{i_{k-1}}, \mathbf{e}_{i_k})$. The pitch of $\mathbf{e}_{i_r}$ is p_{i_r} then the corresponding pitch source sequence is $\langle p_{i_1}, p_{i_2}, \ldots, p_{i_k} \rangle$.

Example 5.2 *In Figure 6,* $\{\mathbf{e}_1, \mathbf{e}_4\}$ *and* $\{\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5\}$ *are the streams. Their linkages are* $\{(\mathbf{e}_1, \mathbf{e}_4), (\mathbf{e}_2, \mathbf{e}_3), (\mathbf{e}_3, \mathbf{e}_5)\}$.

Hit rate and false alarm rate are used to evaluate the stream segregation result of the music in which the natural streams are known. Natural streams, a term borrowed from natural clusters, are pre-defined streams of the music. In other words, the "correct" stream segregation result have already been known. Streams segregated by our algorithm are called output streams, which are compared to the natural streams. A hit is a linkage of an event pair that exists in both the natural stream and the output stream. A false alarm is a linkage that exists in an output stream but not any natural streams. The hit rate and the false alarm rate are defined as follows.

Definition 5.1 (Hit rate)

$$H = \frac{\textit{Number of hits}}{\textit{Number of linkages in natural streams}}$$

Definition 5.2 (False alarm rate)

$$A = \frac{\textit{Number of false alarms}}{\textit{Number of linkages in output streams}}$$

Example 5.3 *Suppose the natural streams in the five-event example in Figure 5 are* $\{\mathbf{e}_1, \mathbf{e}_4\}$, $\{\mathbf{e}_2, \mathbf{e}_5\}$, *and* $\{\mathbf{e}_3\}$ *so the linkages are* $\{(\mathbf{e}_1, \mathbf{e}_4), (\mathbf{e}_2, \mathbf{e}_5)\}$. *The linkages of the output streams are* $\{(\mathbf{e}_1, \mathbf{e}_4), (\mathbf{e}_2, \mathbf{e}_3), (\mathbf{e}_3, \mathbf{e}_5)\}$ *as in Example 5.2. The hit rate* $H = 1/2 = 0.5$ *and the false alarm rate* $A = 2/3 = 0.667$.

Suppose that a subsequence S' matches the query Q. If it is a correct match, all linkages of the corresponding events in S' must be hits; otherwise, the subsequence S' is missed. If the match is a false alarm, at least one linkage of the corresponding events in S' is a false alarm. Assuming that the probability of a hit is independent from the previous hits and the probability of a false alarm is independent from the previous false alarms, we define the probabilities of a hit and of a false alarm for an m-length query in Definitions 5.3 and 5.4 respectively.

Definition 5.3 (Probability of a hit for an m-length query)

$$H_m = (H)^{m-1} \textit{ for } m \geq 2$$

Definition 5.4 (Probability of a false alarm for an m-length query)

$$A_m = 1 - (1 - A)^{m-1} \textit{ for } m \geq 2$$

The probability of a hit for an m-length query is the probability not to miss an m-length query; while the probability of a false alarm for an m-length query is the probability of a matched pattern for an m-length query being a false alarm. In the next section, experiments are performed to verify our proposed solution.

6 Experimental Results

We implement our distance functions and adapt the implementation of the single-link clustering algorithm in [15] for demonstrating the capability and usefulness of our proposed method. We collect real polyphonic music data in the MIDI (Musical Instrument Digital Interface) format, which is the most popular encoding scheme to represent music symbolically.

In the first experiment, we adjust the time weighting factor α and set the pitch weighting factor β to 1 to investigate their effect on the hit rate and the false alarm rate. The MIDI file of Johann Sebastian Bach's two-part Invention No. 1 is collected from [22]. The opening six bars of Invention No. 1 are shown in Figure 7. A two-part Invention is a two-voice keyboard piece, in which the upper voice is played by the right hand and the other by the left. On the facsimile of the title page of the autograph of 1723, Bach wrote that a two-part Invention should be "learned to play cleanly in two parts" [1]. Except the final bar in Invention Nos. 1 and 8, there is at most two musical notes at a time. Thus, the two parts in an Invention are considered as two natural streams in our experiment. The final bars of the Invention Nos. 1 and 8 are removed (No. 8 will be used in the second experiment). In all streams, each musical note has a linkage to its adjacent musical notes. We compare the output streams generated by our method with the natural streams. The speed of Invention No. 1 is the average value over 2 editions, 2 commentaries, and 9 recordings from [1].

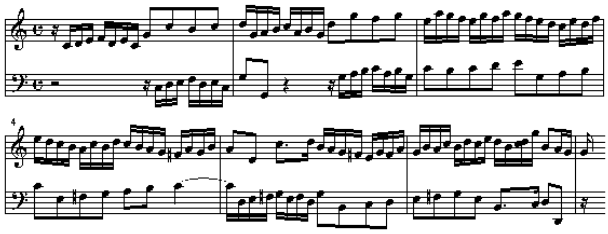
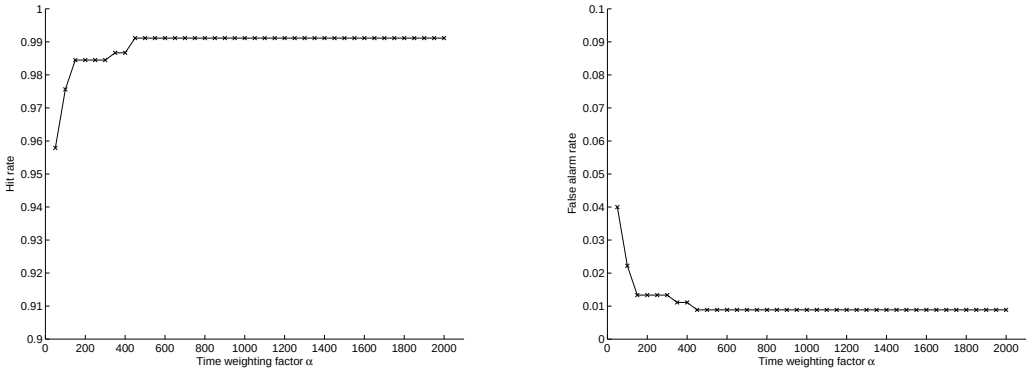


Figure 7: The opening six bars of Bach's Invention No. 1.

The hit rate and the false alarm rate against the time weighting factor are shown in Figures 8(a) and 8(b) respectively. When the time weighting factor increases, the hit rate increases and the false alarm rate decreases and then both of them become stable at 0.991 and 0.009 respectively. In the stable range, we choose the time weighting factor 1000 and the pitch weighting factor 1 arbitrarily and verify these values in the next two experiments. Meanwhile, the clustering result of Invention No. 1 for the time weighting factor 1000 and the pitch weighting factor 1 is depicted in Figure **??**.



(a) (b)

Figure 8: Experiment 1. (a) The hit rate against the time weighting factor α of Bach's Invention No.1. (b) The false alarm rate against the time weighting factor α of Bach's Invention No.1.

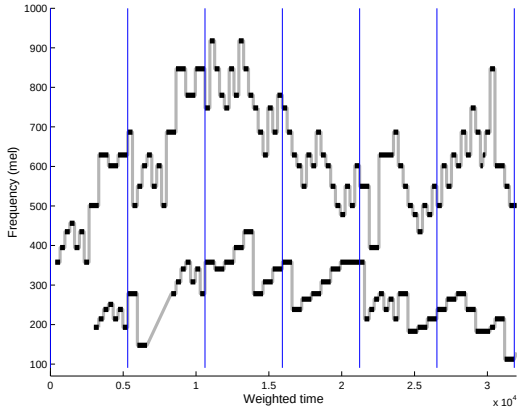


Figure 9: Clustering result of the opening six bars of Invention No. 1 ($\alpha = 1000$; $\beta = 1$). The two continuous lines are the output streams which are the same as the natural streams. The vertical lines are the bar-lines.

In the second experiment, we test our method on all 15 Bach's two-part Inventions collected from [22]. The speed of each Invention is the average value from [1]. We set the time weighting factor α to 1000 and the pitch weighting factor β to 1. In Figure 10, the hit rate and the false alarm rate of the 15 Bach's two-part Inventions are plotted. The average hit rate and the average false alarm rate of the 15 Inventions are 0.986 and 0.015 respectively. The average probabilities of a hit and of a false alarm against the query length are shown in Figure 11. When the length of a query increases, the average

probability of a hit decreases linearly while the average probability of a false alarm increases gradually linearly.

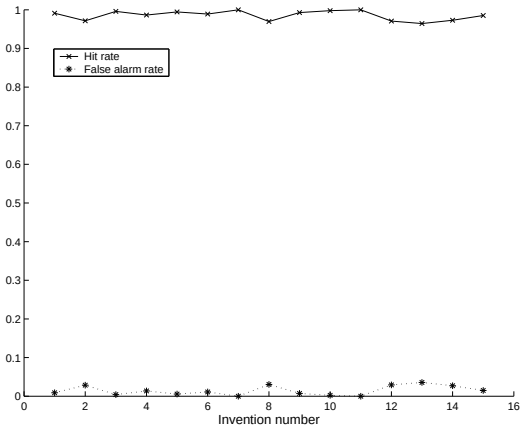


Figure 10: The hit rate and the false alarm rate of the 15 Bach's two-part Inventions

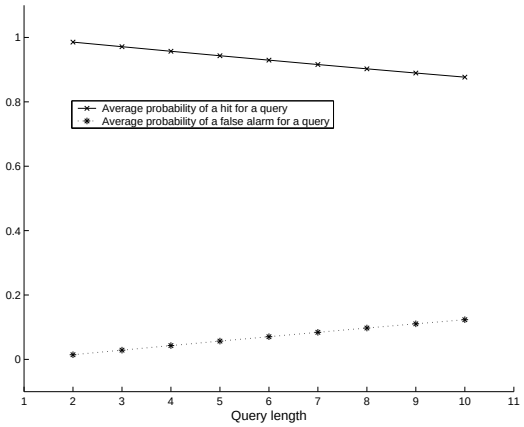


Figure 11: The average probabilities of a hit and a false alarm of the 15 Bach's two-part Inventions against the query length

In the third experiment, we tested whether our method can find the correct linkages of the four streams in the opening two bars of Chopin's Prelude No. 4 in E minor, the piano piece discussed in Section 1. The natural streams are the four streams shown in Figure 2. In all streams, each musical note has a linkage to its adjacent musical notes. The speed is 64.5 crotchet beats per minute, the average value of 6 performances including

Martha Argerich, Jorge Bolet, Shura Cherkassky, Alfred Cortot, Alicia de Larocha, and Maurizio Polini. We also set the time weighting factor α to 1000 and the pitch weighting factor β to 1 as in our previous experiments. The clustering result are depicted in Figure 12. Only four output streams are generated. The hit rate and the false alarm rate are 1 and 0 respectively. No event is misclassified. The probabilities of a hit and of a false alarm against the query length are shown in Figure 13.

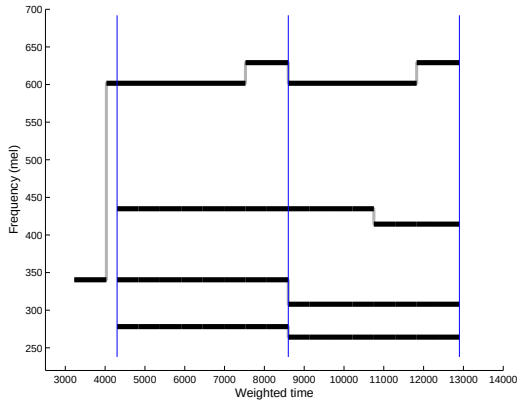


Figure 12: Clustering result of the opening two bars of Chopin's Prelude in E minor ($\alpha = 1000$; $\beta = 1$). The four continuous lines are the output streams which are the same as the natural streams in Figure 2. The vertical lines are the bar-lines.

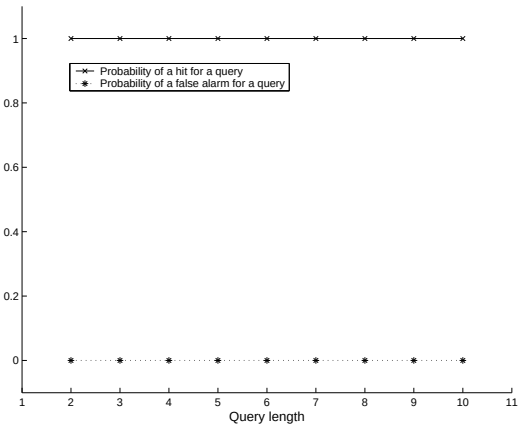


Figure 13: The probabilities of a hit and a false alarm of the opening two bars of Chopin's Prelude in E minor against the query length.

In the fourth experiment, we evaluate the running-time performance of the adapted single-link clustering algorithm. Bach's Invention No. 1 is divided into different lengths. We run our Matlab 6.5 implementation of the adapted single-link algorithm in a Pentium 4 PC with 512MB RAM in Microsoft Windows 2000. The CPU time against the number of events is plotted in Figure 14. Increase in the number of events relates an increase in the CPU time. Although the time complexity of the algorithm is $O(N^3)$, clustering the complete Invention No.1 spends less than 12 seconds.

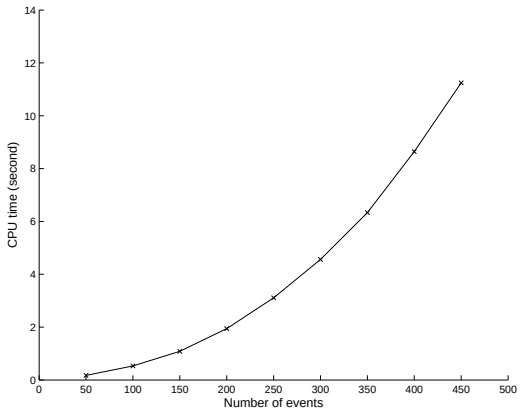


Figure 14: CPU time vs number of events.

7 Conclusion

It is essential to incorporate stream segregation in existing music information retrieval systems in order to improve the quality of retrieval. In this paper, we tackle the stream segregation problem by representing music in the form of events, formulating the inter-event and the inter-cluster distance functions based on the findings in auditory psychology, and applying the distance functions in the adapted single-link clustering algorithm. Supported by the experiments on real music data, our proposed method can successfully identify streams with a high hit rate and a low false alarm rate.

Acknowledgements

We are thankful to Dennis Wu and Eos Cheng for their valuable comments.

References

[1] J. S. Bach. *Inventions and Sinfonias (Two- and Three-Part Inventions).* Alfred Publishing Co., 2nd edition, 1991.

[2] A. S. Bregman. *Auditory Scene Analysis: the Perceptual Organization of Sound.* The MIT Press, 1990.

[3] D. Butler. *The musician's guide to perception and cognition.* Schirmer Books, 1992.

[4] D. Byrd and T. Crawford. Problems of music information retrieval in the real world. *Information Processing and Management*, 38:249–272, 2002.

[5] E. Cambouropoulos, T. Crawford, , and C. Iliopoulos. Pattern processing in melodic sequences: Challenges, caveats and prospects. In *Proc. AISB'99 Symposium on Musical Creativity*, pages 42–47, 1999.

[6] T. Crawford, C.S. Iliopoulos, and R. Raman. String matching techniques for musical similarity and melodic recognition. *Computing in Musicology*, 11:73–100, 1998.

[7] D. Deutsch, editor. *The Psychology of Music.* Academic Press, 2nd edition, 1999.

[8] R. O. Gjerdingen. Apparent motion in music. *Music Perception*, 11:335–370, 1994.

[9] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[10] K. Lemstrom. *String Matching Techniques for Music Retrieval.* PhD thesis, University of Helsinki, Finland, 2000.

[11] K. Lemstrom and S. Perttu. Semex - an efficient music retrieval prototype. In *First International Symposium on Music Information Retrieval (ISMIR 2000)*, Plymouth, Massachusetts, October 23-25 2000.

[12] K. Lemstrom and J. Tarhio. Searching monophonic patterns within polyphonic sources. In *Proc. Content-Based Multimedia Information Access (RIAO'2000)*, volume 2, pages 1261–1279, Paris, France, April 12-14 2000.

[13] F. Lerdahl and R.S. Jackendoff. *A Generative Theory of Tonal Music*. The MIT Press, 1983.

[14] A. Marsden. *Computer Representations and Models in Music*, chapter Modelling the perception of musical voices: A case study in rule-based systems, pages 239–263. Academic Press, 1992.

[15] MathWorks. Statistics toolbox version 4, 2002.

[16] S. L. McCabe and M. J. Denham. A model of auditory streaming. *The Journal of the Acoustical Society of America*, 101(3):1611–1621, March 1997.

[17] R. J. McNab, L. A. Smith, I. H. Witten, and C. L. Henderson. Tune retrieval in the multimedia library. *Multimedia Tools and Applications*, 10(2/3):113–132, 2000.

[18] M. Melucci and N. Orio. Musical information retrieval using melodic surface. In *Proc. ACM International Conference on Digital Libraries*, pages 152–160, 1999.

[19] D. Meredith, G. A. Wiggins, and K. Lemstrom. A geometric approach to repetition discovery and pattern matching in polyphonic music. Presented at Computer Science Colloquium, Department of Computer Science, King's College London, November 21 2001.

[20] D O'Shaughnessy. *Speech Communication*. Addison-Wesley, 1987.

[21] J. Pickens. A survey of feature selection techniques for music information retrieval, 2001.

[22] Mutopia Project. Music listing: J. S. Bach's inventions. http://www.mutopiaproject.org/cgibin/make-table.cgi?Composer=BachJS, 2001.

[23] C. Schachter. *Chopin Studies 2*, chapter The Prelude in E minor Op. 28 No. 4: autograph sources and interpretation, pages 140–161. Cambridge University Press, 1994.

[24] D. Temperley. *The Cognition of Basic Musical Structures*. The MIT Press, 2001.

[25] S. Theodoridis and K. Koutroumbas. *Pattern recognition*. Academic Press, 1999.

[26] A. Uitdenbgerd and J. Zobel. Melodic matching techniques for large music databases. In *Proc. ACM International Multimedia Conference*, pages 57–66, 1999.

[27] L. P. A. S. van Noorden. *Temporal coherence in the perception of tone sequences*. PhD thesis, Eindhoven University of Technology, 1975.