Automated Partitioning of Tonal Music

Bryan Pardo and William P. Birmingham

Artificial Intelligence Laboratory
Electrical Engineering and Computer Science Dept.
The University of Michigan
110 ATL building, 1001 Beal Drive
Ann Arbor, MI 48109
bryanp@umich.edu, wpb@eecs.umich.edu

Abstract

Most research related to automated analysis of music presupposes human partitioning of the input into segments corresponding to significant harmonic or melodic chunks. In this paper, we describe HarmAn, a system that partitions tonal music into harmonically significant segments corresponding to single chords and labels these segments with the proper chord labels. Segment labels are determined through template matching in the space of pitch class with conflict resolution between equally scoring templates resolved through simple preference rules. Our system's results are compared with published results.

Introduction

The analysis of music by computer is a long-standing theme of AI research, as listening to music is clearly an intelligent activity. Listening for the purpose of analyzing the technical aspects of music requires the skills of an expert. One fundamental task in analyzing music, regardless of the approach to be followed, requires breaking the music into the proper "segments," which support later processing. This is analogous to the computer-vision task of finding lines in an image. We are concerned with developing an algorithm for segmentation that makes minimal commitments to (musical) context, thus gaining flexibility and performance.

Analysis of the harmony is central to the technical understanding of any piece of tonal music. In order to label the harmonies in a piece, the analyst must find those places in the music where the harmony changes, and label the resulting segments with the proper chord names. Figure 1 shows a measure of music partitioned into two labeled segments.

In the case of *block chords*, where notes start and end simultaneously, the harmonic changes are obvious. Music, however, is not this simple: chords are often arpeggiated (spread out over time), incompletely stated, and interspersed with myriad *non-harmonic tones* (often stating the melody), in forms such as trills and appoggiaturas. All of these things confound finding the correct changes in the harmony.

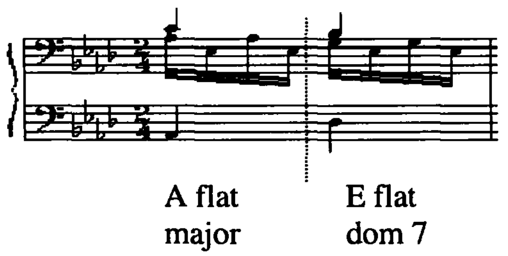

Figure 1: Beethoven, Sonata Pathetique, Op 13, Second Movement, measure 1

Let us introduce some terminology. The harmony of a piece may change each time a note begins or ends. A *partition point* is where the set of pitches currently sounding in the music changes by the starting or ending of one or more notes. P_{all} is the ordered set of all partition points for a piece of music. A *segment* is the interval between any two partition points. A *minimal segment* is the interval between two sequential partition points. Figure 2 illustrates these concepts.

A *partitioning* of a piece is a subset of P_{all} containing at a minimum the first and last elements of P_{all}. A binary number uniquely labels any partitioning of a piece of music. The *i*th bit of the number is "1" if the *i*th partition point in P_{all} is in the partitioning, else the bit is "0". The number of partition points in Figure 2, denoted $|P_{all}|$, is nine.

Figure 1 shows a partitioning of the same music into two segments labeled "A flat major" and "E flat dom 7," respectively. This partitioning is represented by the nine digit number "100010001."

Since a partitioning is uniquely labeled with a binary number of length $|P_{all}|$ and the initial and final bits of this number are always "1", there are $2^{|P_{all}|-2}$ ways to partition any piece of music. Clearly, exhaustive exploration of possible partitionings is intractable for all but the smallest pieces. Thus, any system that performs harmonic analysis must apply a heuristic to reduce the size of the search problem.

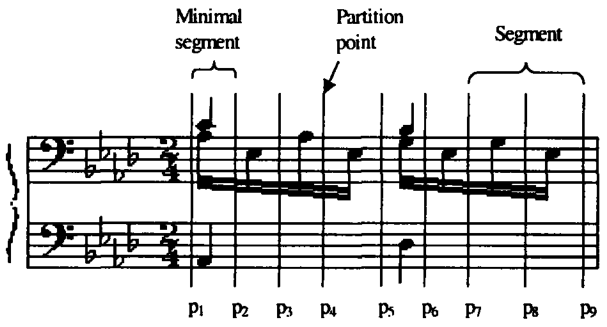

Figure 2: Beethoven, Sonata Pathetique, Op 13, Second Movement, measure 1

Previous research in the area of automated harmonic analysis of music (Winograd 1968; Smoliar 1980; Maxwell 1992; Widmer 1992; Smaill, Wiggins et al. 1993), with the notable exception of recent work by Temperley and Sleator (Temperley and Sleator 1999), has either avoided the issue of partitioning by taking partitioned input, or has been unclear about how partitioning is done.

Finding a good partitioning requires a metric for determining the "goodness" of a partitioning. We do this by generating labels and scores for the segments defined by a particular partitioning. The score of a partitioning is given by the sum of its segments' scores. Partitionings can then be directly compared and the best one selected.

We describe a concise template matching algorithm, related to the work of both Ulrich (Ulrich 1977) and Wakefield (Wakefield 1999; Wakefield and Pardo 1999) that quickly labels a partitioning's segments and generates a score for the partitioning. Our approach decouples labeling a single partitioning from finding the best one. This allows the use of well-known search methods to find a good partitioning. The combination of template matching and our method for search through the space of partitionings produces excellent results. It also yields an approach that is much simpler and easier to extend than previously reported approaches.

Templates for Segment Labeling

The music we are concerned with is based on the pitch classes of the chromatic scale. The chromatic scale and its 12 pitch classes form the basic set of items used to generate the structures associated with most Western music. The common labels for the pitch classes, along with their numeric equivalents, are given in Table 1.

Using the integer representations of the pitch classes and modulo 12 arithmetic, structures such as chords and scales, can be represented as n-tuples representing positive displacements in the space of pitch classes in relation to a root pitch class. These tuples form templates useful for describing musical structures and are related to those used in atonal set theory (Forte 1973), the chromagram (Wakefield 1999; Wakefield and Pardo 1999), and the work of Ulrich (Ulrich 1977).

Table 1: Pitch Class Number and Name Correspondences

C	*C#* *D♭*	*D*	*D#* *E♭*	*E*	*F*	*F#* *G♭*	*G*	*G#* *A♭*	*A*	*A#* *B♭*	*B*
0	1	2	3	4	5	6	7	8	9	10	11

An example of the template representations is the following. Given a root (pitch) class, r, the tuple <0,4,7> represents the pitch class relations to r embodied in a major triad. Letting $r = 7$, this results in a chord given by mod12(r+0, r+4, r+7) = {7,11,2}. Looking at Table 1, it is easy to verify that this corresponds to {G, B, D}, the pitch classes in the G major triad.

Segment Labeling and Scoring

Our method for labeling a segment, embodied in a software system named *HarmAn*, currently uses the templates described in Table 2 to find the best-matching label for the set of notes in that segment. Given a particular segment, the score for a particular template and root combination is calculated using the following steps.

1. Determine the weight of each note by counting the number of minimal segments in which the note is present between the start and end of the current segment.
2. If a note matches a template element, add its weight to the score.
3. If a note does not match any template element, subtract its weight from the score.
4. Subtract one point for each template element not matched by any note.

The first step requires some explanation. Consider the segment from p_1 to p_3 in Figure 2. This segment is comprised of two minimal segments. The "C" at the top of the staff is a quarter note so it is held from p_1 to p_5. It spans both minimal segments in $<p_1, p_3>$ so it has a weight of two. The "A flat" on the top line of the staff holds for only a single minimal segment and has a weight of one.

All combinations of template and root class are scored in order to determine the best (i.e., highest scoring) label for a particular segment. The score of the best label is then used as the score of the segment. Ties between the scores generated for two templates are resolved through the application of the following preference rules.

Prefer major triad to minor triad.

Prefer minor triad to major-minor 7th.

Prefer major-minor 7th to diminished triad.

Prefer diminished triad to augmented triad.

Prefer lower pitch-class numbers.

Chord-quality preferences take precedence over pitch-class number preferences.

Preferences are transitive; for example, major triads are preferred to augmented triads.

Table 2: Common Tonal Structure Representations

Name of tonal structure	Typical written notation, given a "C" root note	Template representation
major triad	C, C Maj, C:I	<0, 4, 7>
minor triad	C min, c:I	<0, 3, 7>
augmented triad	C+, C aug, c:I+	<0, 4, 8>
diminished triad	C dim, c:i°	<0, 3, 6>
major-minor 7th chord	C7, C dom 7, F:V7	<0, 4, 7, 10>

Note that nothing in this approach limits templates to triadic structures. Pentatonic scales, constructions based on fourths or any other structure can be identified simply by introducing a template for the structure in question, and establishing a rule for resolving ties between the new template and existing ones. Thus, extending the system to new tonal structures is easy, and presents no scaling problems. We believe this is also true for atonal structures.

The simplicity and generality of our segment labeling approach is its great strength. We have separated an approach to finding basic structural elements of the harmony from style-specific issues, such as voice-leading rules and rhythmic practices. These may be added as ancillary processing to improve results on difficult cases.

Finding a Good Partitioning

For HarmAn, a performance of a piece of music, M, is represented by a standard MIDI file. The system reads note events from the file and generates the set of all partition points, P_{all} and a set of associated segments, S_m, containing all minimal segments.

P_{good} is the best partitioning found so far. S_{good} is the set of segments defined by P_{good}. Once P_{all} is created, P_{good} is set equal to P_{all}. HarmAn then evaluates each partition point from p_2 through $p_{|P_{all}|-1}$ to determine whether P_{good} could be improved by removing the partition point. Figure 3 describes this in greater detail.

Note that HarmAn has a preference for longer segments over shorter ones. If the score of the union of two adjacent segments is equal to or higher than the sum of their individual scores, the partition point separating them, p_i, is removed from P_{good}.

```
1. Pgood  := Pall
2. Sgood  := Sm
3. i  := 2
4. score<pi-1, pi>    := LabelAndScoreSegment (pi-1, pi)
5. WHILE i < |Pgood|
6.    score<pi, pi+1> := LabelAndScoreSegment (pi, pi+1)
7.    score<pi-1, pi+1>:= LabelAndScoreSegment (pi-1,pi+1)
8.    IF score<pi-1, pi+1> ≥ (score<pi-1, pi> + score<pi,pi+1>)
9.       remove pi from Pgood
10.      remove <pi, pi+1> from Sgood
11.      remove <pi-1, pi> from Sgood
12.      add <pi-1, pi+1> to Sgood
13.   ELSE
14.      i := i + 1
15.   END
16.END
```

Figure 3: Finding a good partitioning

Example: Beethoven, Sonata Pathetique, Second Movement, Measure 1

Figure 2 shows the first measure of the second movement of Beethoven's Sonata Pathetique with its partition points labeled. Consider a harmonic analysis of this measure. The initial partitioning generated by HarmAn contains all partition points and is represented by "111111111."

Table 3 shows the label and score for every segment considered by HarmAn in its analysis of the first measure of the Beethoven example. The vertical key gives the starting partition point for each segment. The horizontal key gives the ending partition point for each segment. Table entries marked "?" show segments not scored by the system.

HarmAn considers partition points in the order in which they occur in the music. In the first round of comparison, the partitioning 111111111 (P_{all}) is compared to partitioning 101111111 ($P_{all} - p_2$). This is done by comparing the scores of segments $<p_1, p_2>$, $<p_2, p_3>$ and $<p_1, p_3>$. The score of $<p_1, p_3>$ is six, and the sum of the scores for $<p_1, p_2>$ and $<p_2, p_3>$ is five. Partition point p_2 is thus removed from P_{good} (i.e., its bit is set to 0).

In Round 2, HarmAn compares partitioning 101111111 with partitioning 100111111 by comparing segments $<p_1, p_3>$, $<p_3, p_4>$ and $<p_1, p_4>$. The unified segment score is higher than sum of the scores of the two component segments so p_3 is discarded from P_{good}.

In Round 3, HarmAn compares $<p_1, p_4> + <p_4, p_5>$ against $<p_1, p_5>$. The values are equal, but the system chooses to remove the partition point due to its preference for long segments.

Round 4 finds HarmAn comparing partitionings 100011111 and 10000111 via $<p_1, p_5>$, $<p_5, p_6>$ and $<p_1, p_6>$. The sum of the scores of $<p_1, p_5>$ and $<p_5, p_6>$ is 15, while the score of the segment $<p_1, p_6>$ is only nine. Removing the partition point p_6 reduces the score, so the partition point remains, and the system moves on to the next round.

Table 3: Segment Labels and Scores

	p2	p3	p4	p5	p6	p7	p8	p9
p1	A flat maj *2 pts*	A flat maj *6 pts*	A flat maj *9 pts*	A flat maj *12pts*	A flat maj *9 pts*	?	?	?
p2	-	A flat maj *3 pts*	?	?	?	?	?	?
p3	-	-	A flat maj *2 pts*	?	?	?	?	?
p4	-	-	-	A flat maj *3 pts*	?	?	?	?
p5	-	-	-	-	G dim *3 pts*	E flat dom7 *6 pts*	E flat dom7 *9 pts*	E flat dom7 *12pts*
p6	-	-	-	-	-	E flat dom 7 *2 pts*	?	?
p7	-	-	-	-	-	-	G dim *3 pts*	?
p8	-	-	-	-	-	-	-	E flat dom7 *2 pts*

In the remaining rounds, HarmAn considers the partition points p_6 through p_8. In each case, the unified segment is chosen over the two separate segments. Thus, p_6 through p_8 are removed from the final partitioning.

Since the initial and final partition points are always in every partitioning, the system is done once it has considered points p_2 through p_8, leaving partitioning 100010001, with the first segment labeled as "A flat major" and the second segment labeled "E flat dominant." Figure 1 shows this partitioning and labeling.

Evaluation of HarmAn

From Table 3, it is clear HarmAn scored only $2|P_{all}|-3$ out of roughly $|P_{all}|^2/2$ possible segments. Similarly, the number of partitionings considered by the system is equal to $|P_{all}|-1$ (two partitionings are compared for partition point p_2 and one new partitioning is considered for each subsequent partition point up through $p_{|P_{all}|-1}$). This number is much smaller than the full $2^{|P_{all}|-2}$ possible partitionings.

We justify this "left-to-right" heuristic in the following way. Music unfolds in time from start to finish. A composer who wishes to write pieces that are decipherable by the listener must create structures that can be understood in a start-to-finish way with limited backtracking to previously heard passages. The expectation for what comes next is determined by what has just been heard, and it is reasonable to assume that the set of likely partitionings under consideration by a human is greatly constrained by what has already transpired in the music. Thus, it is likely that much music is written so that one can partition it in a start-to-finish way with no backtracking.

To test this assumption, HarmAn was altered to search through the partition points in start-to-finish order, reverse order and in random order. The different search orders were then compared on a test set of Bach Inventions. As expected, start-to-finish searching produced the best results.

In order to compare our results to existing work on automated analysis of harmony, we analyzed a set of pieces by Bach, Beethoven, and Schubert used in other papers. In all cases, HarmAn performed analysis that was considered reasonable by human experts and was comparable to or better than that of existing systems. Details of the experiments performed with HarmAn can be found at http://musen.engin.umich.edu/. What follows is a comparison of HarmAn's analysis of an example passage from Beethoven's Sonata Pathetique to the analysis done by what we consider to be one of the best current systems.

Comparison to Temperley and Sleator

Temperley and Sleator (Temperley and Sleator 1999) have created a system that does beat finding in a manner influenced by Lerdahl and Jackendoff (Lerdahl and Jackendoff 1983). Beats are then used to help determine the partitioning of the piece into time spans, which are labeled as likely chords. A root is then chosen for each segment. Roots are chosen to prefer giving the same root name to successive chords if possible, and to prefer root names related by a fifth, otherwise. Their system is limited to labeling the root of each segment rather than providing the full chordal spelling. The full history of preceding root names is used, along with the intervals present in the current segment, to determine the choice of the root name of each segment. The system is described as a set of preference rules, and is strongly tied to an explicit model of functional tonal harmony in its approach to analysis.

Figure 4 shows the results of analysis of the first eight measures of the second movement of Beethoven's Sonata Pathetique. The root names generated by Temperley and Sleator are on the line labeled "T&S" (Temperley and Sleator only reported the analysis of the first five measures in their paper). The chordal names returned by HarmAn are on the line labeled "HarmAn." As can be seen from the figure, HarmAn successfully captured the correct chord roots and qualities in this passage, with the possible exception of the first half of the final measure.

The Temperley and Sleator system did about as well as HarmAn on root finding in this example, although HarmAn also correctly identifies chord quality. Both systems agree on root spellings through the first four measures, diverging in the fifth measure. Their system finds four roots in this measure, namely "G," "B flat," "E flat," and "A flat." HarmAn reports only two. Interestingly, HarmAn gets closer to capturing the actual harmonic rhythm of this measure even though it does not explicitly represent rhythm. HarmAn also correctly labels measures six seven.

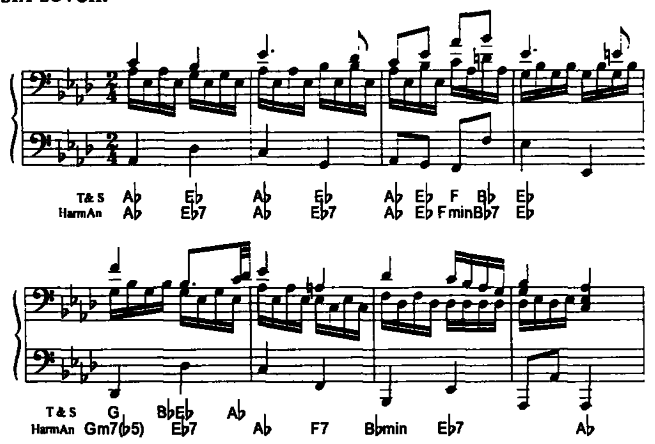
T& S Ab Eb Ab Eb Ab Eb F Bb Eb
HarmAn Ab Eb7 Ab Eb7 Ab Eb Fmin Bb7 Eb
T & S G BbEb Ab
HarmAn Gm7(b5) Eb7 Ab F7 Bbmin Eb7 Ab

Figure 4 : Beethoven, Sonata Pathetique, Op 13, Second Movement, measures 1 through 8

Measure eight is a possible problem. The bass notes on the first two beats of the measure change to "A flat," while the upper voices continue to spell out an "E flat 7" chord. HarmAn's weighting system interprets this as a continuation of the "E flat 7" from the previous measure, with the "A flats" considered as non-harmonic notes. While this is an acceptable solution, a music theorist might treat the whole final measure as a statement of "A flat" with the upper voices functioning as a suspension of the previous harmony, which is resolved in the third beat of the measure. We argue that this interpretation involves semantic knowledge of a particular style (a context), and is thus outside the scope of what our system is designed to address.

Summary

The work described in this paper provides a set of concise and efficient algorithms for context-free harmonic analysis of tonal music and provides a good first approximation of the harmonic structures in a typical piece of tonal music. HarmAn does this using a relative small set of rules, which do not require an understanding of the tonal context, nor any metrical information to perform the analysis. The results achieved by this system compare well to those achieved by much more complex systems reported in the literature. Furthermore, HarmAn is easily extendable to new harmonic structures.

References

Forte, A. 1973. *The Structure of Atonal Music*, Yale University Press.

Lerdahl, F. and Jackendoff, R. 1983. *A Generative Theory of Tonal Music*. Cambridge, Mass, MIT Press.

Maxwell, J. H. 1992. An Expert System for Harmonizing Analysis of Tonal Music. *Understanding Music with AI: Perspectives on Music Cognition*. Camgridge, Mass, MIT Press.

Smaill, A. and Wiggins, G. et al. 1993. Hierarchical Music Representation for Composition and Analysis. *Computers and the Humanties* **27**: 7-17.

Smoliar, S. 1980. A Computer Aid for Schenkerian Analysis. *Computer Music Journal* **4**(2).

Temperley, D. and Sleator, D. 1999. Modeling Meter and Harmony: A Preference-Rule Approach. *Computer Music Journal* **23**(1): 10-27.

Ulrich, J. W. 1977. The Analysis and Synthesis of Jazz by Computer. *Proceedings from the 5th IJCAI.*

Wakefield, G. H. 1999. Chromagram Visualization of the Singing Voice. *International Workshop on Models and Analysis of Vocal Emissions for Biomedical Applications*, Firenze, Italia.

Wakefield, G. H. and Pardo, B. 1999. Signal Classification using Time-Pitch-Chroma Representations. *The Intl. Symp. on Opt. Sci., Eng., and Instr., SPIE'99*, Denver, Colorado, USA.

Widmer, G. 1992. Perception Modeling and Intelligent Musical Learning. *Computer Music Journal* **16**(2).

Winograd, T. 1968. Linguistics and the Computer Analysis of Tonal Harmony. *The Journal of Music Theory* **12**: 2-49.