Modeling Meter and Harmony: A Preference-Rule Approach

David Temperley[*] and Daniel Sleator[†]
[*]School of Music
Weigel Hall
Ohio State University
Columbus, Ohio 43201, USA
temperley.1@osu.edu
[†]School of Computer Science
Carnegie-Mellon University
5000 Forbes Avenue
Pittsburgh, Pennsylvania 15213, USA
sleator+@cs.cmu.edu

Introduction

Computational music analysis is an important project for a number of reasons. From a psychological point of view, systems for performing musical processes may shed light on how human listeners perform such processes, just as computational work in vision and problem solving has led to important insights in those domains. While recent research has revealed much about listeners' mental representations of music (Dowling and Harwood 1986; Butler 1992), there is much to be learned in this area. There are practical applications to automatic music analysis as well. Many kinds of music-related tasks that computers might perform—for example, producing a score from raw pitch data, detecting probable errors in a score, searching a melodic database for matches to a query, generating an accompaniment for a melody, or performing statistical analysis of musical styles for the purposes of automatic composition—require an ability to process music and extract various kinds of structure and information from it. Extracting this information proves to be a highly complex task.

In this article, we present a computational system for analyzing metrical and harmonic structure. The system is designed for Western tonal music, particularly art music of the "common-practice" period. As input, the program takes a list of notes with pitch, on-time, and off-time; it produces a representation showing a metrical structure (consisting of several levels of beats) and a harmonic structure (consisting of a partitioning of the piece into segments, each labeled with a root). (There are important reasons for combining the metrical and harmonic systems into a single program, which we will explain.) Our approach is based on *preference rules*. Preference rules are criteria for selecting an analysis of a piece out of many possible ones. The preferred analysis is the one which, on balance, best satisfies the rules. The current article presents an overview of our project; more information is available at the World Wide Web site http://www.link.cs.cmu.edu/music-analysis. The complete source code for our system (written in C) is available at this site, and we also provide a number of sample input files and a utility program for generating input files from MIDI files. The site also contains information about recent work that we were unable to include here.

Both metrical analysis and harmonic analysis have been addressed before in computational music studies. The problem of harmonic analysis has received surprisingly little attention. The few attempts that have been made (notably by Winograd [1968] and Maxwell [1992]) have had limited success. The most important limitation is that they are unable to handle cases where the notes of a chord are not stated fully and simultaneously, such as arpeggiations, incomplete chords, and unaccompanied melodies. They also require information that is not generally present in listening, such as key signatures, pitch spellings, and rhythmic values. (Temperley [1997] has reviewed this work.) More work has been done on metrical analysis (Longuet-Higgins and Steedman 1971; Steedman 1977; Chafe, Mont-Reynaud, and Rush 1982; Povel and Essens 1985; Allen and Dannenberg 1990; Lee

Computer Music Journal, 23:1, pp. 10–27, Spring 1999
© 1999 Massachusetts Institute of Technology.

1991; Rosenthal 1992; Parncutt 1994). In this area too, previous systems have been limited in both the inputs they handle, and the representations they produce. Some can handle only quantized inputs (Longuet-Higgins and Steedman 1971; Steedman 1977; Povel and Essens 1985; Lee 1991; Parncutt 1994); some consider only rhythmic information, without considering pitch in any way (Longuet-Higgins and Steedman 1971; Povel and Essens 1985; Allen and Dannenberg 1990; Lee 1991; Parncutt 1994); some produce only a single level of beats (Povel and Essens 1985; Parncutt 1994); and all of these systems are limited to monophonic music. The current system attempts to overcome these limitations.

Quite apart from the issue of performance, the system presented here differs from most other harmonic and metrical analysis systems in an important way. For the most part, the systems mentioned above are not preference-rule systems. Rather, they might better be called *procedural* systems: they are best described in terms of the procedure they follow, rather than the output they produce. Certainly, these systems reflect principles about the kinds of metrical structures that are desirable, like having strong beats on long notes. However, it is not usually possible to describe the output of the system as one that satisfies some set of criteria. (Two exceptions are the systems of Povel and Essens and Parncutt, which could be viewed as simple preference-rule systems.) With preference-rule systems, by contrast, the output can be described simply as the one that best satisfies the preference rules. To put it another way, the preference rules themselves provide a kind of high-level description of what the system is doing, which is not usually available in procedural systems.

The conceptual simplicity of preference-rule systems is one of their attractive features. However, they present a computational problem: how to find the optimal analysis of a piece out of a huge number of possibilities. We will propose a solution to this problem that seems to have quite broad applicability to musical preference-rule systems.

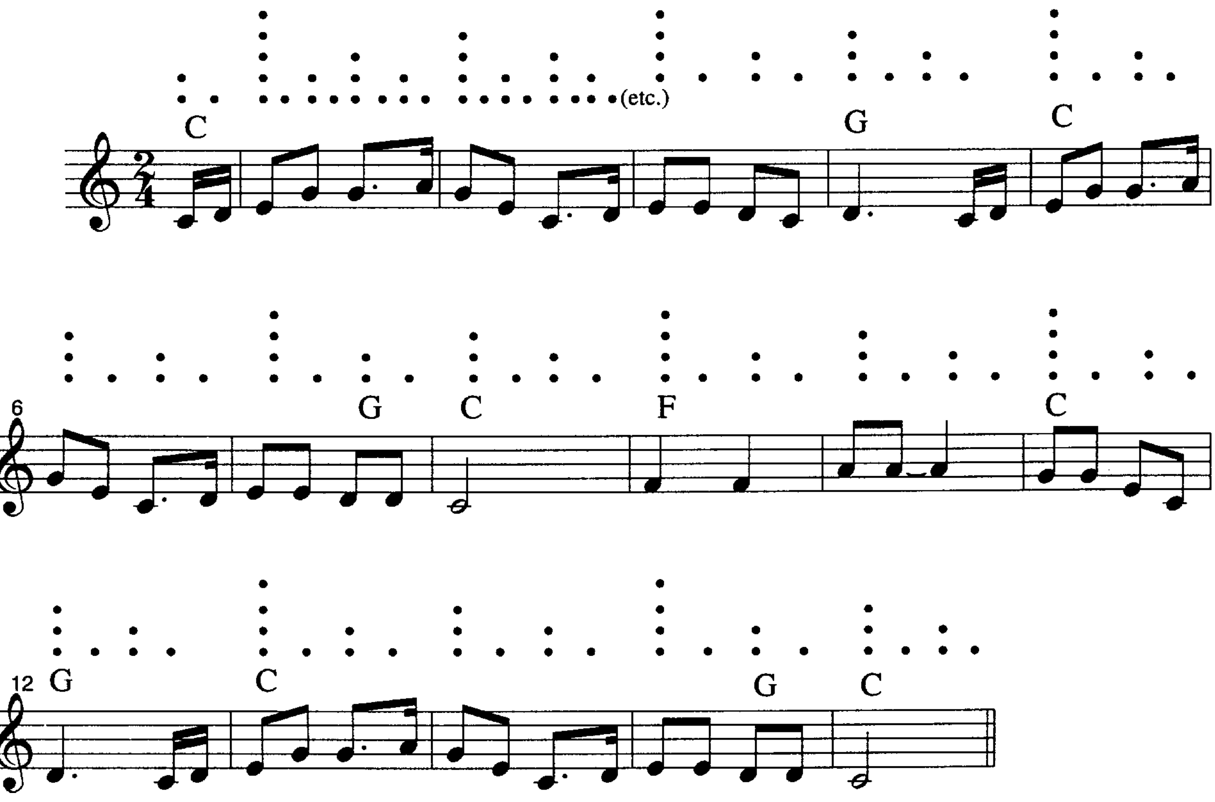
We begin by describing our two preference-rule systems. With each system, we begin with an informal description, and then explain the way it is formalized and implemented. We also discuss the procedure we have devised for solving the search problem posed above. We examine several examples of the program's output, pointing to some virtues and failings. Finally, we consider some possible directions for improvement, and discuss some important general advantages of the preference-rule approach.

Meter: The GTTM Model

Before we begin, a word is needed about the input to the program. The input we assume is a "note list," giving the pitch (in integer notation, middle C = 60) and the on-time and off-time (in milliseconds) of a series of notes. It is perhaps easiest to think of this as a two-dimensional representation with pitch on one axis and time on the other, similar to a piano roll. Each event is represented as a line segment on this plane, the length of the segment corresponding to the note's duration. The program requires no information beyond this: in particular, it does not require other information commonly available in scores, such as bar lines, key signatures, rhythmic notation, and the spellings of pitches.

While our approach to meter builds on various earlier attempts, it is most closely related to the theory of Lerdahl and Jackendoff, as presented in *A Generative Theory of Tonal Music* (hereafter called *GTTM*) (1983). Lerdahl and Jackendoff propose a theory of meter, stated as a set of well-formedness (or hard-and-fast) rules and preference rules. The preference rules in *GTTM* are stated informally, and the authors do not discuss how they might be formalized or implemented. Lerdahl and Jackendoff propose that a metrical structure consists of several levels of equally spaced beats (where beats are points in time, not necessarily events). Every second or third beat at one level is a beat at the next level up. (By convention, the level with the fastest beats is called the lowest level.) In music notation, the time signature of a piece usually indicates several levels of the metrical structure. For example, a piece in 6/8 time has one level of eighth-note beats; every third beat at that level

Figure 1. "Oh Susannah," showing the correct metrical and harmonic structure. Each row of dots indicates a level of the metrical structure; each dot indicates a beat on the onset of the note below. Each chord symbol represents a chord span, beginning on the note below and extending to the beginning of the following chord span.



forms a dotted-quarter level of beats; and every second beat at that level forms a dotted-half (or "one-measure") level. In addition, there may also be one or two levels above the measure, so-called hypermetrical levels.

Even assuming that a metrical structure must involve several levels of equally spaced beats, one must still determine the duple or triple relationships between levels, the tempo (the time intervals between beats), and the placing of the beats relative to the music. For this purpose, Lerdahl and Jackendoff posit a set of metrical preference rules, stating the criteria whereby listeners infer the correct structure. Consider Figure 1, the traditional American melody "Oh Susannah." The correct metrical structure is shown above the staff. (The metrical and harmonic analysis shown here is the one produced by our program, with one difference that we will explain.) The most important rule is that beats (especially higher-level, or "strong" beats) should whenever possible coincide with the onsets of events. Second, there is a preference for strong beats to coincide with longer events. In "Oh Susannah," for example, this favors placing quarter-note beats on even-numbered eighth-note beats (the second, fourth, and so on), since this aligns them with the dotted eighth notes. Similarly, this rule favors placing half-note beats on odd-numbered quarter notes, since this aligns the long note in measure 4 with a half-note beat. We state these rules as follows (our wording differs slightly from that in *GTTM*):

> **Event rule**—prefer a structure that aligns beats with event onsets
> **Length rule**—prefer a structure that aligns strong beats with onsets of longer events

Note that the preferred metrical structure is the one that is preferred on balance; it may not be preferred at every moment of the piece. For example,

Figure 2. Mozart's Sonata, K. 332, first movement, measures 1–5.

Figure 3. A case where duration affects the perceived length of notes.

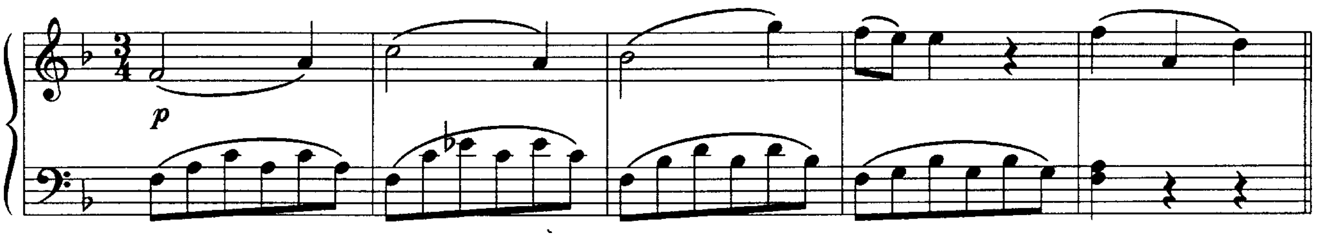


Figure 2

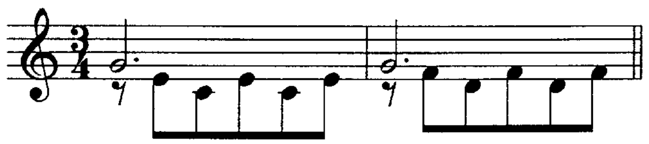


Figure 3

the second A in measure 10 is a long note on a fairly weak beat, thus violating the length rule, but on balance, this structure is still the preferred one.

When we turn to polyphonic music, several complications arise. Consider Figure 2, the opening of Mozart's *Sonata,* K. 332. The correct metrical structure is indicated by the time signature; why is this the perceived structure? Clearly, the eighth-note level is determined by the event rule. The quarter-note level could also be explained by the event rule: on certain eighth-note beats we have two event onsets, not just one. This brings up an important point about the event rule: the more event onsets at a time point, the better a beat location it is. Regarding the dotted-half-note level, one might suppose that it was due to the long notes in the right hand. This raises the question of what is meant by the "length" of an event. The actual length of events is available in the input, and this could be used. However, this is problematic. In the Mozart excerpt, the long notes in the right hand would probably still seem long (and hence good candidates for strong beats) even if they were played staccato. Alternatively, one might assume that the length of a note corresponds to its *inter-onset interval* (*IOI*), which is the time interval between the note's onset and the onset of the following note. (This is the approach taken by most previous meter-finding programs.) This approach works fairly well in monophonic music (although even there it encounters problems, as we will show); however, it is totally unworkable in polyphonic music. In the Mozart piece, the IOI of the first right-hand event is only one eighth note: it is followed immediately by a note in the left hand. Intuitively, what we want is the IOI of a note within that line of the texture: we call this the *registral IOI.* However, separating the events of a texture into lines is a highly complex task that we will not address here. Our solution to this problem, which is crude but fairly effective, is to define the registral IOI of an event as the inter-onset interval to the next event within a certain range of pitch: we adopt the value of nine semitones. However, it sometimes proves useful to use duration as well. In Figure 3, the fact that the melody note is held makes it clear that it is in a separate line from the lower notes, and hence is truly long, despite its short registral IOI. Taking all this into account, we propose a measure of an event's length that is used for the purpose of the length rule: the length of a note is the maximum of its duration and its registral IOI.

Lerdahl and Jackendoff's theory neglects one extremely important aspect of meter. The *GTTM* rules state that beats at the "tactus" level and immediately higher levels must be exactly evenly spaced. The tactus is the most salient level of meter, usually corresponding to the main "beat" of the music in colloquial terms. In actual performance, however, beats are of course not exactly evenly spaced at any level. There may be deliberate fluctuations in timing, as well as errors and imperfections. This is a crucial point, because it means that we cannot simply infer the metrical structure at the beginning of a piece and extrapolate it metronomically through the rest of the piece (even if we had the ability to do that): we must continuously adjust the meter to fit the music that is heard. (There is no doubt that listeners can and do make these adjustments; in general, we have little difficulty relocating the beat after a fermata or ritard, or even a complete change in meter or tempo.) In principle, accommodating this into Lerdahl and Jackendoff's theory is straightforward: we simply make the requirement for regularity of beats at each level into a preference rule, rather than a well-formedness (hard-and-fast) rule. We state this rule as follows:

> **Regularity rule**—prefer beats at each level to be maximally evenly spaced

Maximizing the regularity of each level thus becomes a flexible constraint, to be balanced against the other preference rules.

The regularity rule brings up an important feature of our system. In the past, computational analysis of rhythm has generally been divided into two problems. One is quantization: the rounding off of durations and time points in a piece to multiples of a common beat (see, for example, Desain and Honing 1992). The other is metrical analysis: imposing higher levels of beats on an existing lower level. Models that assume a quantized input (such as Lerdahl and Jackendoff's) are really only addressing the second problem. However, an important recent realization of music artificial intelligence has been that quantization and meter finding are really part of the same process. In imposing a low level of beats on a piece of music, marking the onsets of events, one is in effect identifying their position and duration in terms of integer values of those beats. Most notably, Rosenthal (1992) proposed a model that accomplishes both low-level quantization and medium-level meter finding. Our system is similar in this respect, performing both quantization and meter finding in a single process.

The three rules discussed above—the event rule, the length rule, and the regularity rule—form the core of the meter program. It remains to be explained how they are formalized and implemented. The basic idea is straightforward. The system must consider all possible analyses of a piece, where an analysis is simply a well-formed metrical structure, consisting of several rows of beats, as outlined above. Each metrical preference rule assigns a numerical score to each analysis, indicating how well the analysis satisfies that rule. The preferred analysis is the one with the highest total score.

For now, let us consider the simpler situation of deriving a single row of beats whose time intervals may vary over a broad range, say 400–1,600 msec, the typical range for the tactus level. We will return later to the problem of adding other levels. An analysis, then, is a row of beats imposed on a given piece. The input representation is first quantized into very short segments of 35 msec, which we call *pips* (the value of 35 msec was simply found to be optimal through trial and error). Every event onset and offset is rounded to the nearest pip; beats also may occur only at the start of a pip.

An analysis can then be evaluated in the following way. Each pip containing a beat is given a numerical score indicating how many events have onsets at that point. This score is also weighted by the length of the events starting there. In summing these *note scores* for all the pips, we have a numerical representation of how well the analysis satisfies the event and length rules. Each pip containing a beat is also given a score indicating how evenly spaced it is in the context of the previous beats in that analysis. There are various ways that this could be quantified, but we have found a very simple method that works well: the regularity of a beat B_n is simply given by the absolute difference between the interval between B_n and B_{n-1} and that between B_{n-1} and B_{n-2}. This *beat-interval score* therefore acts as a penalty: a representation is preferred in which the beat-interval scores are minimized. The preferred tactus level is the one whose total note score and (negative) beat-interval score is highest. One complication here is that simply defining the note score for a beat level as the sum of the note scores for all its beats gives an advantage to analyses with more beats. Thus, we weight the note score of each beat by the square root of its beat interval (the interval to the previous beat). The square root was chosen simply because it is a convenient function whose growth rate is between a constant (which would be too small) and a linear function (which would be too large).

It is because of the beat-interval scores that the scores must be computed in terms of entire analyses. The best beat location within (for example) a 1-sec segment depends in part on the beat-interval penalty for different pips, but this penalty depends on the location of previous beats, which in turn depends on their beat-interval scores (and their note scores), and so on. In theory, then, all possible analyses must be considered to be sure of finding the highest-scoring one overall. However, the number of possible analyses increases exponentially with the number of pips in the piece. There is therefore a search problem to be solved of finding the correct analysis without actually generating them all; we discuss later how we solve this problem.

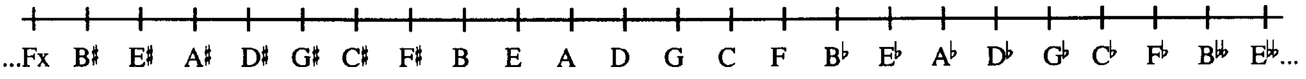
The preferred tactus level for a piece, then, is the one that best satisfies the three preference rules, quantified in the way just described. It remains to be explained how the other levels are derived. The program begins with the tactus level; it then generates two levels above the tactus and two below. Five levels prove to be sufficient for the great majority of pieces. We have adopted the convention of calling the tactus level 2, the upper levels 3 and 4, and the lower levels 1 and 0. The generation of the upper levels is straightforward. Level 3 is generated in exactly the same way as level 2, with the added stipulation that every beat at level 3 must also be a beat at level 2, and exactly one or two level-2 beats must elapse between each pair of level-3 beats. Level 4 is then generated from level 3 in the same manner. There is no explicit requirement for regularity in terms of the relationship between levels (duple or triple), but this tends to emerge naturally, since there is pressure on each level to be regular in itself. The lower levels are slightly more complicated. At level 1, a possible duple or triple division is generated for each level-2 beat. The best choice between duple and triple, and the exact placing of the lower-level beats within the tactus beat, is determined using the same preference rules described above. However, there is also a penalty for switching from duple to triple from one tactus beat to another. In this way, other things being equal, there is a preference for maintaining either duple or triple division once it is established.

A Model of Harmony

Another essential aspect of musical structure is harmony. Elsewhere one of us has proposed a model for harmonic analysis (Temperley 1997); this is the basis for the system we present here. Like the metrical program, the main input to the harmonic program is simply a "piano roll": a list of events, with a time and duration for each event (although metrical information is also required, as we explain). The program produces a representation of segments, which we call *chord spans*, labeled with roots. This is somewhat different from conventional harmonic analysis or "Roman-numeral analysis": whereas a Roman-numeral analysis represents each root relative to the current key, the current program simply labels roots in absolute terms.

The model we employ is, again, a preference-rule system, giving a set of criteria for choosing the preferred analysis out of all the possible ones. A possible harmonic analysis is simply a complete segmentation of the piece, where each segment is labeled with a root. Again, "Oh Susannah" provides a simple illustration of the program's rules. What we would consider the correct analysis is indicated by the root symbols below the staff. (Unlike metrical structure—which is usually indicated by the score of a piece—the correct harmonic structure for a piece is to some extent a matter of opinion. While there are certainly cases where people disagree about the correct analysis for a piece, there is a great deal of agreement as well.) Assuming the first segment consists of the first three measures (plus the opening pickup), why does C seem like the correct choice of root here? The most obvious reason is that C, E, and G are all present, and these are chord tones of C major. One question is how we know which notes in the segment are chord tones, as opposed to ornamental tones; we will return to this. But even once this is known, the chord tones in a segment often do not uniquely identify a chord. For example, measures 9–10 contain F and A; these could imply either an F-major chord or a D-minor chord, although F major seems preferred. To capture such intuitions, we stipulate that roots are preferred

Figure 4. The line of fifths.


...Fx B♯ E♯ A♯ D♯ G♯ C♯ F♯ B E A D G C F B♭ E♭ A♭ D♭ G♭ C♭ F♭ B𝄫 E𝄫...

that result in certain pitch-root relationships, but some relationships are preferred to others. We express this in the following rule:

> **Compatibility rule**—prefer roots that result in certain pitch-root relationships. The following relationships are preferred, in this order: 1, 5, 3, ♭3, ♭7, ♭5, ♭9, ornamental

In measures 9–10, a root of F is thus preferred over D: given the pitches F and A, F results in pitch-root relationships of 1 and 3, whereas D results in 5 and ♭3.

The compatibility rule says that an event that does not have a chord-tone relationship with the current root is "ornamental." However, not just any event can be ornamental: some events are better ornamental tones than others. If one considers the most common types of ornamental tones—passing tones, neighbor tones, suspensions, and appoggiaturas—one finds that they all share a common characteristic: they are closely followed by another event a half-step or whole-step away in pitch. In addition, we have found that there is a preference for ornamental tones to be metrically weak, although sometimes they are metrically strong. (This of course requires access to metrical structure; we will return to this in a moment.) In a scale passage, for example, the metrically strong notes are more likely to be heard as chord tones. We express these intuitions as follows:

> **Ornamental dissonance rule**—in labeling events as ornamental, prefer events that are (1) closely followed by another event a half-step or whole-step away, and (2) metrically weak

According to this rule, the D in the pickup to measure 1, the A at the end of measure 1, and the D at the end of measure 2 are all good ornamental dissonances, since all are closely followed stepwise. However, the G in measure 2 (for example) is not closely followed stepwise. If this event were treated as ornamental—that is, if a root were chosen for that segment of which G was not a chord tone—the ornamental dissonance rule would be severely violated.

Now consider measure 4. According to the compatibility rule, D would be the preferred root for this segment (since the root D results in a pitch-root relationship of 1 with pitch D). Why is G more preferred? Intuitively, G is a "closer" harmony to C than D is, and should therefore be preferred in such cases. Here we introduce a simple spatial model: the *line of fifths*, similar to the circle of fifths, but extending infinitely in either direction as shown in Figure 4. Choosing a root for a segment is therefore a matter of mapping it onto a point on the line of fifths. Given this model, we can say that there is a preference for choosing roots so that roots of nearby segments are close together on the line. This leads to a third rule:

> **Harmonic variance rule**—prefer roots such that roots of nearby chord spans are close together on the line of fifths

The line of fifths raises the issue of spelling. Much work in music theory (especially atonal theory) and music cognition has assumed that pitches are categorized into "pitch classes," such that pitches one or more octave apart are members of the same class. However, in conventional tonal theory and music notation, distinctions are made between different spellings of the same pitch; e.g., A♭ versus G♯. We call these *tonal pitch classes* (TPCs), as opposed to the twelve *neutral pitch classes* of atonal theory. It is our view that these spelling distinctions are an important aspect of tonal harmony, and must be represented. (For a discussion of this issue, see Temperley 1997.) This adds a further task for the program: as well as dividing the piece into segments labeled with roots, it must also choose a spelling label for each pitch event, hence mapping it onto the line of fifths. The main criterion for doing this is a simple one, analogous to the harmonic variance rule above:

Pitch-variance rule—prefer spellings for pitch events such that nearby events are close together on the line of fifths

The representation of pitch spellings—which we call the *TPC representation*—is closely integrated with the harmonic representation. In the first place, the compatibility rule, which stipulates the preferred pitch-root relationships, refers to tonal pitch classes, not neutral ones. That is, if a segment contains A♭-C-E♭, the preferred root for that segment is A♭, not G♯. However, the program seeks the overall representation—of both roots and pitches—that maximally satisfies all the rules, the harmonic rules and the pitch-variance rule. In some cases, harmonic considerations may force a pitch spelling that would be less preferred given the pitch-variance rule alone. In this way, harmonic considerations "feed back" to influence pitch spelling.

A final harmonic rule concerns the division of the piece into chord spans. We have generally assumed chord spans of one measure or more, but why do we assume this? Alternatively, one could posit a separate chord span for each note. In general, there seems to be a preference for longer chord spans, or at least an avoidance of very short ones. However, there is another principle involved as well. Consider the G segment in measure 4. Why not begin this chord span two notes earlier, on the previous D (thus making the C a neighbor tone)? The reason is that there is a preference for starting chord spans on strong beats of the metrical structure. This has the added effect of avoiding very short chord spans; since strong beats are never very close together, if a chord span is very short, either that chord span or the following one must begin on a weak beat. We express this as follows:

Strong-beat rule—prefer to start chord spans on strong beats

This important rule explains the motivation for incorporating the harmonic and metrical programs into a single program. Since harmonic analysis requires metrical information, it is useful to be able to use the output of the metrical program as input to the harmonic program. There is a problem here, however, as metrical analysis sometimes requires harmonic information. We will return to this issue below.

The implementation approach we take with the harmonic program is similar to that taken with the metrical program. The harmonic program considers all possible analyses of the entire piece; in this case, a possible analysis is simply a complete segmentation of the piece into chord spans labeled with roots. (A spelling must also be chosen for each pitch event. The way this is handled computationally is rather complex, and is not discussed here.) Again, we begin by dividing the piece into short segments. Here we can use the already-generated metrical structure. Recall that any chord span beginning on a very weak beat will get an extremely high penalty. It seems intuitively plausible to exclude altogether chord-span boundaries that do not coincide with beats at all. Therefore, we divide the piece into segments based on the lowest level of the metrical structure (usually on the order of 100–300 msec), and stipulate that chord-span boundaries can only occur at these segment boundaries. A root is chosen for each segment, and a chord span then emerges as any group of contiguous segments with the same root. (Since the number of possible roots is infinite, we arbitrarily limit it to a range of 48 roots on the line of fifths, and stipulate that the first root must be in the range from F♯ to D♭.)

In evaluating a harmonic analysis, each preference rule assigns a numerical score to each segment, indicating how well it satisfies that rule. Beginning with the compatibility rule, we give the segment a positive score for each event (or part of an event) in the segment that has a chord-tone relationship with the root given by the analysis; more preferred relationships result in higher scores. Regarding ornamental tones, each event that is not a chord tone of the root receives a penalty, depending on how "good" it is as an ornamental tone, that is, how closely followed it is by the next event a whole step or half step away, and how metrically weak it is; better ornamental tones receive lower penalties. Turning to the strong-beat rule, we assign a penalty to each segment whose root is different from the previous segment, de-

Figure 5. A hypothetical search table for the simplified harmonic algorithm.

Segment	1	2	3	4	5
Root					
C	5	3 (8,C)	1 (9,C)	1 (12,G)	0 (15,E)
C#	0	1 (4,C)	0 (7,G)	0 (11,G)	2 (17,E)
D	2	0 (3,C)	0 (7,G)	2 (13,G)	0 (15,E)
E♭	0	4 (7,C)	0 (7,G)	0 (11,G)	0 (15,E)
E	1	0 (3,C)	5 (12,G)	5 (17,E)	3 (20,E)
F	4	3 (7,F)	2 (9,F)	1 (12,G)	1 (16,E)
F#	0	0 (3,C)	0 (7,G)	0 (11,G)	0 (15,E)
G	3	6 (9,C)	4 (13,G)	2 (15,G)	3 (18,E)
A♭	1	0 (3,C)	2 (9,G)	3 (14,G)	0 (15,E)
A	1	1 (4,C)	2 (9,G)	3 (14,G)	6 (21,E)
B♭	0	1 (4,C)	1 (8,G)	0 (11,G)	2 (17,E)
B	1	0 (3,C)	2 (9,G)	3 (14,G)	0 (15,E)

pending on the strength of the beat at that point; weaker beats result in higher penalties. The harmonic variance rule is slightly more complex. For each segment, we take the mean position on the line of fifths of all previous roots in the piece, weighted for recency, so that more recent segments affect it more. We then look at the distance of the current segment's root from this "center of gravity." This provides a measure of how close the current segment's root is to previous roots. We then assign each segment a penalty based on this value. (Incidentally, one advantage of using the line of fifths as a space for roots and pitch classes—rather than a circular space, for example—is that it permits this easy way of calculating spatial proximity.)

If only the compatibility rule and the ornamental dissonance rule were involved, finding the preferred analysis would be easy, since the root for each segment could simply be chosen in isolation. What makes it difficult are the variance and strong-beat rules. Because of these, the preferred root for a segment depends on its context. As with the metrical program, then, there is a search problem to be solved of finding the highest-scoring analysis out of all possible ones.

The Search Procedure

Our algorithms for finding optimal solutions to the search problems described above are based on dynamic programming (Cormen, Leiserson, and Rivest 1990). The procedures for the harmonic and meter preference-rule systems are similar, and are summarized in this section.

We explain our approach using a somewhat simplified example. Imagine a harmonic algorithm that uses only two of the rules discussed here: the compatibility rule, as proposed above, and a "no-change" rule, which simply penalizes all harmonic changes (similar to the strong-beat rule, except disregarding meter). Assume also, for simplicity's sake, that there are only twelve roots. Now imagine a piece consisting of a series of segments. Each segment has a compatibility score for each root, reflecting how compatible the pitches of the segment are with the root. We can represent these scores in a table, as shown in Figure 5. The numbers in each cell represent the compatibility scores for each segment with each root. (Ignore the numbers in parentheses for the moment.) Let us assume that the no-change rule imposes a penalty of 2 points. Each analysis is thus a path along the table, with one step in each column; each analysis receives a compatibility score, which is the total of the scores in the squares it steps in, minus 2 points for each move from one row to another. The object is to find the highest-scoring path.

Let us begin by considering just the first two segments, S_1 and S_2. We calculate the best analysis of these two segments by simply considering all 144 possibilities: each root for S_1, combined with each root for S_2. (The best analysis is C-G, with a score of 5 + 6 – 2 = 9.) Now we proceed to the third segment, S_3. The score for a three-segment analysis can be viewed as the score for the first two segments (already calculated), plus the compatibility score and no-change penalty (if any) for the third segment. It might appear that we have to calculate scores for all three-segment analyses to find the best one. However, there is a major shortcut we can take here. Consider the analyses of S_1–S_2 that end in root F. We have already calculated the scores for these, and found that F-F scores the highest. When adding on a third segment, then, we need only consider the highest scoring of the analyses ending in each root at S_2. There is no reason that any of the others would ever be preferred. S_3 only cares about what the root was in the previous segment; anything prior to that is irrelevant. Thus we proceed as follows. For each root at S_2 we record the score for the highest-scoring analysis ending at that root, along with the S_1 root that the analysis entails. These "best-so-far" scores, along with the best previous root, are shown in parentheses in the table. At S_3, we consider each root at S_3 combined with each best-so-far analysis ending at each root at S_2; for each root at S_3 we choose a new best-so-far analysis, and record the score and the S_2 root it entails. We repeat this process through the entire piece. In this way, each square in each column points back to a square in the previous column. When we get to the end of the piece, we look at the best-so-far scores for the roots in the final segment, and choose the highest score. By tracing this analysis back through the table, we can then reconstruct the highest-scoring possible analysis for the entire piece. In this case, it is C-G-E-E-A.

At any point in a piece, one of the roots will have the highest-scoring best-so-far analysis, and that will represent the preferred analysis at that point. For example, at S_3 the preferred root is G. But consider S_4; here the preferred root is E, and it points back to a root of E at S_3. In this way, the system naturally handles the "garden-path" effect: the phenomenon of revising one's initial interpretation of a segment based on what follows.

This is the procedure we use for both the metrical and harmonic programs. With the meter program, we can imagine the columns of the table representing pips. Each pip has a note score; rows of the table represent beat intervals, i.e., the time to some previous pip. Consider the simplified problem of deriving a single beat level. A beat level is a path through the table, placing beats at certain locations. (Unlike the example above, a beat analysis only steps in certain columns of the table.) There is a penalty for switching rows (i.e., beat intervals) from one step to the next. The preferred analysis is the one that hits the highest note scores with the fewest and smallest shifts in beat interval. So the problem becomes that of filling the entry in the table for a specific pip and beat interval. This is done by looking back to some previous pip (determined by the beat interval) and choosing the best beat interval to use at that prior pip. When evaluating this choice, we have available the current beat interval and the hypothetical previous beat interval, so we can assign the appropriate beat-interval penalty. Once the end of the table is reached, the highest score represents the preferred analysis overall, and this can be traced back through the table.

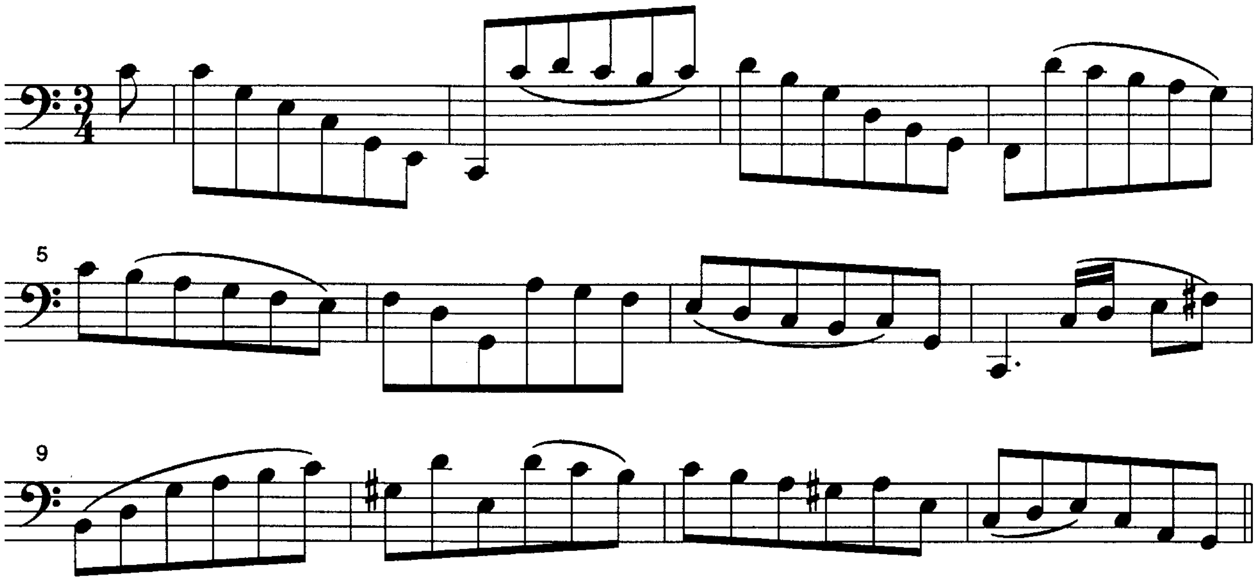
Regarding the harmonic program, the simple algorithm described above can handle the compatibility rule, the ornamental dissonance rule, and the strong-beat rule (which penalizes changes of harmony, depending on the strength of the beat). The dynamic programming table at each segment only needs to be a one-dimensional array; that dimension is the choice of root of the segment. To incorporate the harmonic variance rule and the pitch-variance rule (determining the spellings of pitches), things get a lot more complex, and we end up with a four-dimensional table (rather than a one-dimensional table) to fill in. Space limitations prevent us from giving the details here.

In practice, the computation described above is intractable, because the four-dimensional table gets too large. We handle this by pruning the table for a given segment immediately after the table for that segment is constructed. We simply keep only those elements that are within some constant of the highest score in the table. We have found a value for this constant that maintains a reasonable speed for the system without compromising accuracy.

Examples

We have tested the program on a number of pieces and sections of pieces. These include both quantized files generated from scores (e.g., Finale files), and unquantized files generated from live performances on a MIDI keyboard. Most are pieces from the common-practice (Bach to Brahms) era, mainly piano pieces; there are also a number of unaccompanied melodies. All of the input files we have

Figure 6. Bach's Suite for Violoncello *No. 3, Courante, measures 1–12.*



tested (including those discussed below) are available at our World Wide Web site. Interested readers can judge the program's performance for themselves. In this section, we discuss representative examples of the program's output, pointing to some strengths and weaknesses along the way. Some of the problems we discuss here have recently been addressed, and are detailed on our World Wide Web site.

Both the metrical and harmonic programs involve a number of parameters. The weight of each preference rule relative to the others must be specified. Many rules also involve internal parameters: for example, in the compatibility rule, the relative preference of different pitch-root relationships. We have simply adjusted these parameters on a trial-and-error basis. After many tests and adjustments, we have found a set of values that seems to produce generally good results. (The parameters can easily be adjusted; users of the program may wish to experiment with this.)

Our first example is the opening of the *Courante* from Bach's *Suite for Violoncello in C Major*. The score is shown in Figure 6; the program's output is in Figure 7.

A word of explanation is needed about the format of the output. Each line of the output represents a beat at the lowest level of meter. (Recall that these are treated as indivisible units by the harmonic program.) At the far left is the time point at which that unit starts, given in milliseconds. Remember that these time points have been quantized to pips—units of 35 msec. To the right of that is a series of xs indicating the levels of beats present at the time point. In both this example and the next, the lowest level of meter in the program's output is omitted to save space; thus the tactus level is the second from the left. Next is the root of the segment shown according to its position on the line of fifths, and finally the pitches contained in the segment are listed (by their chosen spellings) and represented graphically on the line of fifths. Each pitch event is only indicated by name in the segment in which it begins. Graphically, each pitch event is represented by a + symbol in the segment where it begins, and by a | symbol in subsequent segments where it is present. If more than one event of a particular TPC is present, this is indicated with 2, 3, etc., as appropriate.

The Bach example demonstrates several nice aspects of the program. The metrical structure is correct here, with the possible exception of the highest level. (The input here is quantized, generated from a score rather than a live performance.)

Figure 7. The program’s output for the Bach passage shown in Figure 6.

```
M.     Meter         Harmonic Rep                     TPC Rep
no.
        0  x          C                      <C  >                +
1     210  x x x      C                      <C  >                +
      385  x          C                      <C  >                   +
      595  x x        C                      <E  >                            +
      770  x          C                      <C  >                +
      980  x x        C                      <G  >                   +
     1155  x          C                      <E  >                            +
2    1365  x x x x    C                      <C  >                +
     1540  x          C                      <C  >                +
     1750  x x        C                      <D  >                      +
     1925  x          C                      <C  >                +
     2135  x x        C                      <B  >                               +
     2310  x          C                      <C  >                +
3    2520  x x x         G                   <D  >                      +
     2695  x             G                   <B  >                               +
     2905  x x           G                   <G  >                   +
     3080  x             G                   <D  >                      +
     3290  x x           G                   <B  >                               +
     3500  x             G                   <G  >                   +
4    3675  x x x x       G                   <F  >             +
     3885  x             G                   <D  >                      +
     4060  x x           G                   <C  >                +
     4270  x             G                   <B  >                               +
     4445  x x           G                   <A  >                         +
     4655  x             G                   <G  >                   +
5    4830  x x x      C                      <C  >                +
     5040  x          C                      <B  >                               +
     5215  x x        C                      <A  >                         +
     5425  x          C                      <G  >                   +
     5600  x x        C                      <F  >             +
     5810  x          C                      <E  >                            +
6    5985  x x x x       G                   <F  >             +
     6195  x             G                   <D  >                      +
     6370  x x           G                   <G  >                   +
     6580  x             G                   <A  >                         +
     6790  x x           G                   <G  >                   +
     6965  x             G                   <F  >             +
7    7175  x x x      C                      <E  >                            +
     7350  x          C                      <D  >                      +
     7560  x x        C                      <C  >                +
     7735  x          C                      <B  >                               +
     7945  x x        C                      <C  >                +
     8120  x          C                      <G  >                   +
8    8330  x x x x    C                      <C  >                +
     8505  x          C                      <   >                |
     8715  x x        C                      <   >                |
     8890  x          C                      <C  >                +
     9100  x x        C                      <E  >                            +
     9275  x          C                      <F# >                                  +
9    9485  x x x         G                   <B  >                               +
     9695  x             G                   <D  >                      +
     9870  x x           G                   <G  >                   +
    10080  x             G                   <A  >                         +
    10255  x x           G                   <B  >                               +
    10465  x             G                   <C  >                +
10  10640  x x x x                  E        <G# >                                        +
    10850  x                        E        <D  >                      +
    11025  x x                      E        <E  >                            +
    11235  x                        E        <D  >                      +
    11410  x x                      E        <C  >                +
    11620  x                        E        <B  >                               +
11  11795  x x x                 A           <C  >                +
    12005  x                     A           <B  >                               +
    12180  x x                   A           <A  >                         +
    12390  x                     A           <G# >                                        +
    12565  x x                   A           <A  >                         +
    12775  x                     A           <E  >                            +
12  12985  x x x x               A           <C  >                +
    13160  x                     A           <D  >                      +
    13370  x x                   A           <E  >                            +
    13545  x                     A           <C  >                +
    13755  x x                   A           <A  >                         +
    13930  x                     A           <G  >                   +
```

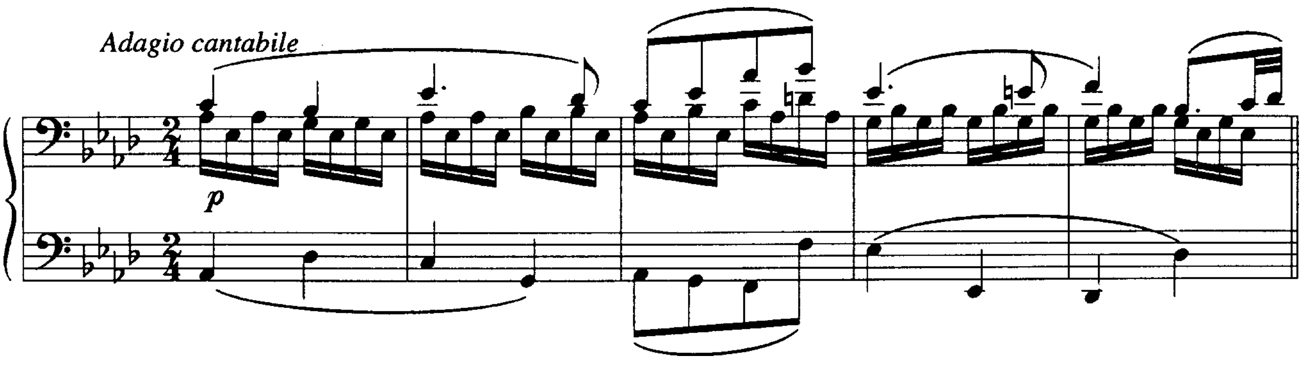


Figure 8. Beethoven's Sonata, *Op. 13 ("*Pathetique*") II, measures 1–5.*

The meter is somewhat subtle, since (with the exception of measure 8) there is a note on every eighth-note beat, and all the notes are the same length. One important cue seems to be the low notes at the beginning of measures 2 and 4. The concept of registral IOI is important here; although these notes are not actually long, they are not closely followed by another note in the same register, which makes them seem long. Turning to the harmonic structure, this is (in our opinion) exactly correct in this passage. In a piece such as this, of course, it is essential for the program to be able to handle harmonies in which the notes are stated successively rather than all at once. Note also the importance of meter in determining the harmony. It is this that allows the program to decide where chord segments begin and end. Without it, for instance, the first G segment might begin two notes earlier.

Our second example, the opening of the slow movement of Beethoven's Op. 13 (*"Pathetique"*) Sonata, tests an important aspect of the meter program: its ability to handle fluctuations in tempo. The score is shown in Figure 8, and the program's output is in Figure 9.

Unlike the previous example, this example is from a performance played on a MIDI keyboard. If one consults the time data in the left column, it can be seen that the performer (David Temperley) makes considerable variations in the tempo. For convenience, the intervals between tactus beats are shown at the far left; these are not normally displayed by the program. (The program considers the tactus to be the sixteenth-note level here; the eighth-note level would perhaps be a better choice.) Again, the very lowest level of the metrical structure is not shown in this example, thus the tactus level is the second from the left. Among the features of the expressive timing here are a slight slowing down at the beginnings and endings of measures; a significant acceleration in the first three quarters of measure 3 (the mean tactus interval here is 437 msec, as opposed to a mean of 491 msec for the whole passage); and, most notably, a substantial ritard at the end of measure 4, with two consecutive beats of 630 msec. Altogether, the tactus intervals fluctuate between 420–630 msec. The program is able to handle all of these fluctuations, maintaining the correct metrical structure throughout. The chords here greatly help the program to identify where the strong beats are. Unaccompanied melodies played with rhythmic freedom often give the program trouble.

The next example, the opening of Schubert's *Moment Musical* #6 (see Figure 10), demonstrates some strengths and weaknesses of the harmonic program. In this case, rather than showing the output, we simply show the program's analysis as chord symbols on the score; each chord symbol indicates a chord span beginning on the note below. The analysis of the opening phrase could be improved in several respects. Labeling measure 1 as D♭ is odd; it would make more sense to consider measures 1–2 as part of a single B♭ chord. Similarly, measure 3 would probably be better labeled an A♭ chord with a long appoggiatura. One general weakness of the program is that it has no knowledge of pedals. Measure 7 would probably best be

Figure 9. The program's output for the Beethoven passage shown in Figure 8.

```
               Meter          Harmonic Rep                   TPC Rep
Tactus
intervals
               0  x x x x Ab                      <Ab Ab C      >
             280  x       Ab                      <             >
595          595  x x     Ab                      <Eb           >
             840  x       Ab                      <             >
490         1085  x x x   Ab                      <Ab           >
            1295  x       Ab                      <             >
455         1540  x x     Ab                      <Eb           >
            1750  x       Ab                      <             >
455         1995  x x x x    Eb                   <Db Bb G      >
            2240  x          Eb                   <             >
490         2485  x x        Eb                   <Eb           >
            2730  x          Eb                   <             >
525         3010  x x x      Eb                   <G            >
            3220  x          Eb                   <             >
455         3465  x x        Eb                   <Eb           >
            3710  x          Eb                   <             >
525         3990  x x x x Ab                      <Ab Eb C      >
            4235  x       Ab                      <             >
525         4515  x x     Ab                      <Eb           >
            4795  x       Ab                      <             >
560         5075  x x x   Ab                      <Ab           >
            5320  x       Ab                      <             >
490         5565  x x     Ab                      <Eb           >
            5810  x       Ab                      <             >
490         6055  x x x x    Eb                   <Bb G         >
            6300  x          Eb                   <             >
490         6545  x x        Eb                   <Eb           >
            6790  x          Eb                   <             >
525         7070  x x x      Eb                   <Db Bb        >
            7315  x          Eb                   <             >
490         7560  x x        Eb                   <Eb           >
            7805  x          Eb                   <             >
490         8050  x x x x Ab                      <Ab Ab C      >
            8260  x       Ab                      <             >
420         8470  x x     Ab                      <Eb           >
            8680  x       Ab                      <             >
455         8925  x x x      Eb                   <Eb Bb G      >
            9135  x          Eb                   <             >
455         9380  x x        Eb                   <Eb           >
            9590  x          Eb                   <             >
420         9800  x x x x           F             <Ab F  C      >
           10010  x                 F             <             >
455        10255  x x               F             <Ab           >
           10465  x                 F             <             >
420        10675  x x x         Bb                <Bb F  D      >
           10920  x             Bb                <             >
490        11165  x x           Bb                <Ab           >
           11410  x             Bb                <             >
490        11655  x x x x    Eb                   <Eb Eb G      >
           11900  x          Eb                   <             >
525        12180  x x        Eb                   <Bb           >
           12390  x          Eb                   <             >
455        12635  x x x      Eb                   <G            >
           12845  x          Eb                   <             >
455        13090  x x        Eb                   <Bb           >
           13300  x          Eb                   <             >
455        13545  x x x x    Eb                   <Eb G         >
           13790  x          Eb                   <             >
525        14070  x x        Eb                   <Bb           >
           14315  x          Eb                   <             >
490        14560  x x x      Eb                   <Fb G         >
           14805  x          Eb                   <             >
525        15085  x x        Eb                   <Bb           >
           15400  x          Eb                   <             >
630        15715  x x x x                 G       <Db F  G      >
           16030  x                       G       <             >
630        16345  x x                     G       <Bb           >
           16555  x                       G       <             >
420        16765  x x x                   G       <G            >
           16975  x                       G       <             >
455        17220  x x           Bb                <Bb           >
           17430  x             Bb                <             >
455        17675  x x x x    Eb                   <Db Bb G      >
           17885  x          Eb                   <             >
420        18095  x x        Eb                   <Eb           >
           18305  x          Eb                   <             >
455        18550  x x x      Eb                   <G            >
           18760  x          Eb                   <             >
455        19005  x x     Ab                      <Eb C         >
           19285  x       Ab                      <Db           >
```

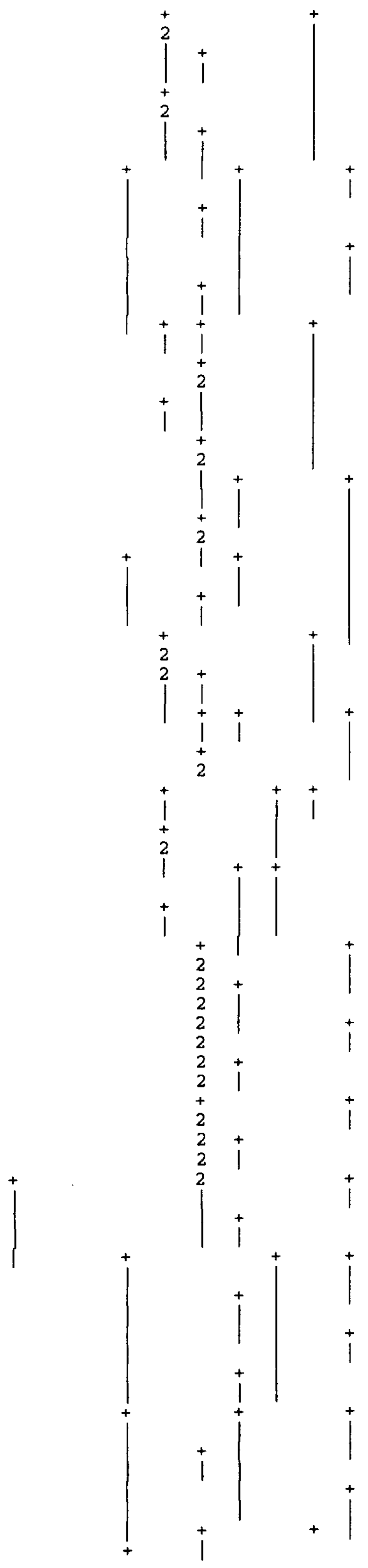
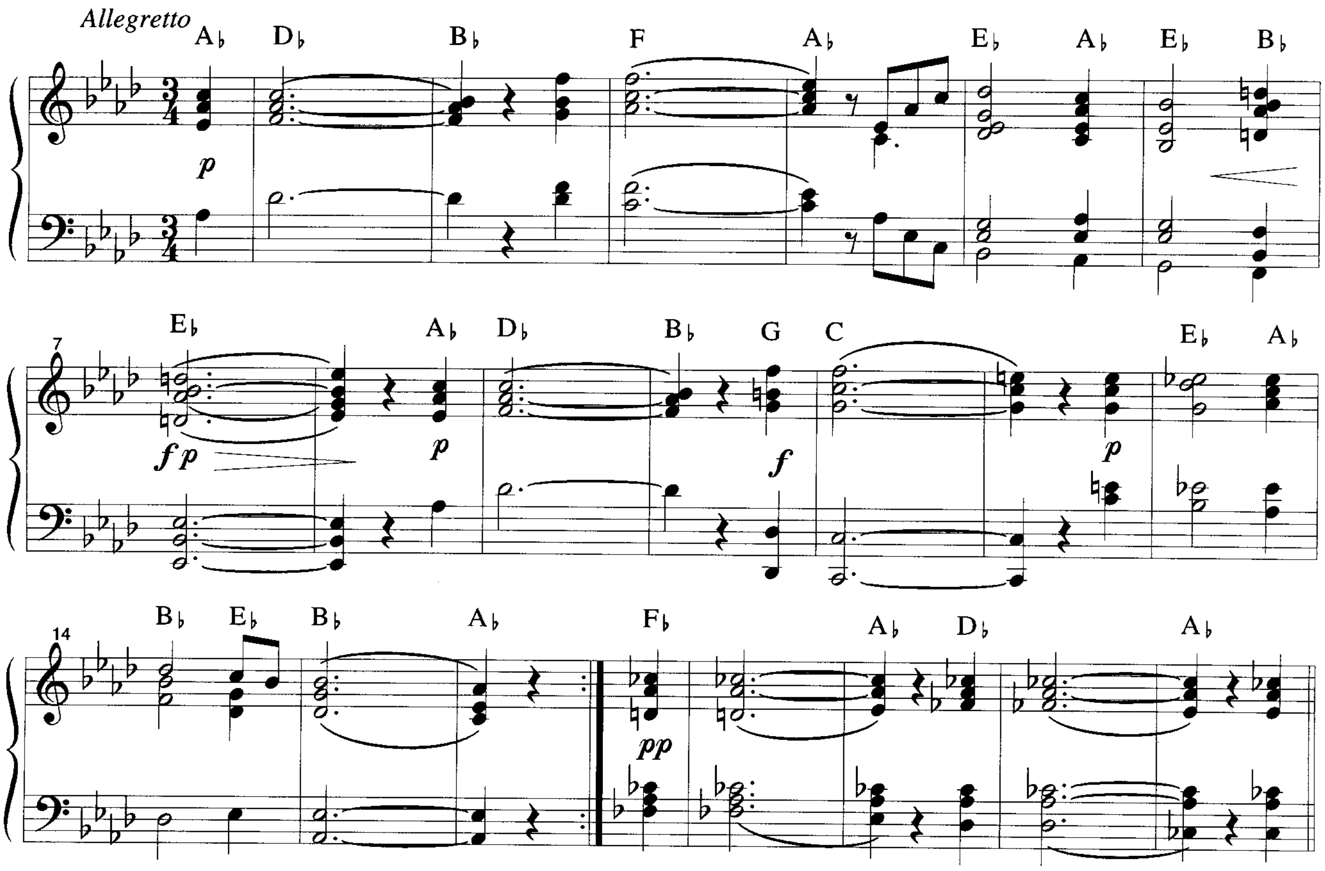


Figure 10. Schubert's Moment Musical *No. 6 (Op. 94), measures 1–20, showing the program's harmonic analysis.*



analyzed as a B♭7 chord over an E♭ pedal. Similarly, measure 15 is an E♭7 chord over an A♭ bass. Being ignorant of such possibilities, the program must try something else. Measure 7 is analyzed, reasonably, as an E♭ chord with several appoggiaturas, but measure 15 is bizarrely labeled as a B♭ chord. Turning to the more chromatic harmonies, the program analyzes the last beat of measure 10 as 1-3-♭5-♭7 with root G—a "French sixth"—just as it should. However, the "German sixth" in measures 16–17 is incorrectly labeled as an F♭ dominant seventh; moreover, the D is misspelled as an E𝄫 (the only spelling error in this passage). This error brings up another general failing of the program: it has no knowledge of voice leading. In measures 16–17, the fact that the D/E𝄫 resolves to an E♭ would normally require a D spelling rather than E𝄫. The program's ignorance of voice leading results in a fair number of spelling mistakes. For the most part, however, the pitch-variance rule and the feedback from harmonic considerations ensure the correct spellings. For example, the program is able to correctly identify the B and E in measures 10–12 and the C♭ and F♭ in measures 16–18.

The harmonic program has other weaknesses. It has no knowledge of several common types of ornamental tones, such as anticipations and escape tones. It sometimes misses common progressions like II-V-I, analyzing them in some other strange way. (As listeners, perhaps we give a "bonus" to such progressions, due to their familiarity and structural importance.) The output of the program is also somewhat unsatisfactory, in that it indicates only the roots of chords, without further information such as mode (major or minor), extension (triad or seventh), and inversion. We hope to address these problems.

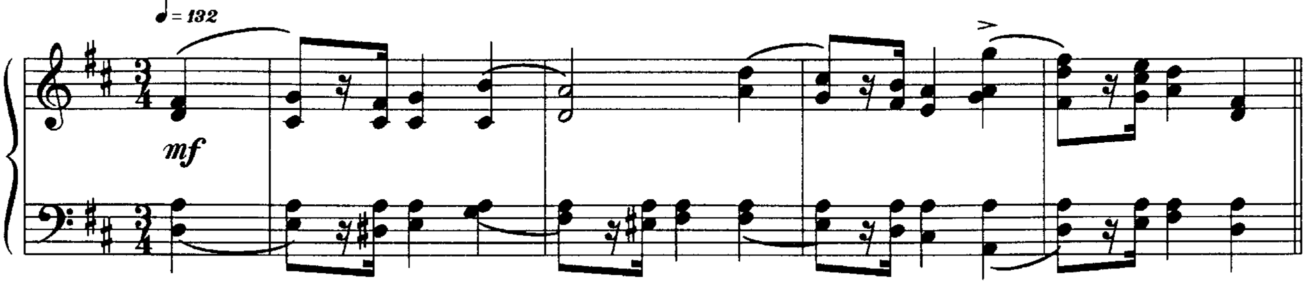
Further Issues

We have explored several possible improvements of the metrical program. In its basic form, as described above, the program finds and maintains the correct tactus level on a large majority of pieces. Its performance on the lower levels is mostly quite good, although it sometimes misses notes in rapid scale passages. (We do not want it to hit *every* note. Some notes are *extrametrical*, that is, notes that would be notated in small note heads in a score: rolled chords, grace notes, trills, Chopinesque ornamental flourishes, and so on. But the program does not always correctly distinguish the metrical notes from the extrametrical ones.) The performance on the upper levels is weaker, especially on level 4. Frequently the program correctly identifies level 4 as duple, which it usually is, but chooses the incorrect phase (this occasionally happens with level 3 as well). In "Oh Susannah," for example (see Figure 1), the program judges even-numbered measures as metrically strong at level 4; the same error occurs in the Bach *courante* (see Figure 7). The reason for this is clear: there are often long notes at the ends of phrases, which makes the program prefer them as strong, although they are often weak at higher metrical levels. The real solution to this problem would be to incorporate what Lerdahl and Jackendoff call "grouping structure." Grouping structure is a hierarchy of segments, with lower-level segments representing motives and larger segments representing phrases and sections. As Lerdahl and Jackendoff note, grouping affects meter in that there is a preference to locate strong beats near the beginning of groups. However, getting a computer to recognize grouping boundaries proves to be a very difficult problem, and our preliminary efforts have been unsuccessful. Instead, we have adopted a cruder solution. At level 4, the program ignores the length rule and simply prefers beat locations that hit the maximum number of event onsets. In addition, we give a slight bonus at level 4 for placing a level-4 beat on the first level-3 beat of the piece (rather than the second or third), thus encouraging a strong level-4 beat near the beginning of the piece. This fix improves performance somewhat, but incorporating grouping structure would clearly be more satisfactory.

Making use of the harmonic analysis is another approach to improving the performance of the metrical program on the higher levels. Consider the Schumann piece shown in Figure 11; the tactus here is the quarter note. Our standard metrical program will identify the level above the tactus as triple, but out of phase with the notated meter (the first quarter note is metrically strong). Perceptually, the important cue here seems to be harmony; there are clear chord changes on the second and fifth tactus beats, which favors strong beats there (again, this factor is noted by Lerdahl and Jackendoff). The idea, then, is to let the harmonic analysis influence the metrical analysis by favoring strong beats at changes of harmony. This presents a serious chicken-and-egg problem, however, since meter is crucial as input to harmony. One solution would be to compute everything at once, optimizing over both the metrical and harmonic rules, but we have not yet found an efficient way of doing this. Another solution, which we are currently exploring, is to first run the piece through the harmonic program, generating a provisional harmonic analysis, then run the output of that through the meter program, which is now modified to prefer strong beats at points of harmonic change, and finally run this output through the harmonic program again to generate the final harmonic analysis. This technique can be used to improve the metrical analysis on a number of pieces, including the Schumann piece discussed here. The software included in our distribution can be run in this manner.

Another factor that is involved in meter is loudness. It would be quite easy to incorporate loudness as a factor in our system; however, we do not believe it is a major determinant of metrical structure (see Rosenthal 1992 for discussion). Finally, the factor of parallelism should be mentioned: the preference to align metrical levels with patterns of

Figure 11. Schumann's Op. 15 (Kinderscenen), *No. 2.*



repetition. This is undoubtedly a factor in meter, but we have not yet addressed the major task of incorporating it. (For an interesting attempt to handle parallelism, see Steedman 1977.)

While the main purpose of our system is to generate satisfactory analyses for pieces, it has some further interesting features that deserve mention. One is its handling of real-time listening. As noted earlier, the system's analysis of one segment of a piece—both metrical and harmonic—is the one entailed by the optimal analysis of the piece as a whole: the analysis of a segment may be affected by both its prior and subsequent context. Since the system processes pieces in a left-to-right fashion, its initial analysis of a segment may be revised based on what happens afterwards. In this way, the program may shed light on subtle nuances of listening, such as the garden-path effect mentioned earlier. This is an emergent feature of the system that seems to warrant further exploration.

Another interesting feature of the system relates to its evaluation of possible analyses. In searching for the optimal analysis of a piece, the system is in effect assigning numerical scores to various analyses of the piece and comparing them. When it is finished, what it produces is not only an optimal analysis, but a numerical score for that analysis. More importantly, it produces a series of scores for segments of the analysis, and for the different rules, indicating how well each segment satisfies each rule. In some cases, an analysis of a segment may be found that satisfies all the rules reasonably well. In other cases, however, the best analysis available may be quite low scoring on one or several of the rules, and may (in comparison to other segments) be quite low scoring overall. These scores may reflect interesting aspects of music and musical experience. To take just two examples, a segment featuring wide harmonic leaps on the line of fifths—a sudden move to a remote harmony—would score poorly on the harmonic variance rule; a passage featuring harmonic changes on very weak beats, or very rapid harmonic motion, would score poorly on the strong-beat rule. These are devices in tonal music that can be used to create tension or instability. In this way, the numerical scores produced by the program can be seen as indicators of the degree of tension of musical passages. Like the garden-path effects observed earlier, nothing special has to be done to produce these numerical scores. They arise naturally as a result of the system's search for the optimal analysis.

In several ways, then, preference-rule systems provide powerful models of aspects of music cognition. While we have focused here on meter and harmony, it seems that preference-rule systems would be applicable to other aspects of musical structure as well. One clear candidate is grouping structure: the segmentation of a piece into motives, phrases, and sections. Lerdahl and Jackendoff offer a preference-rule system for grouping in *GTTM*, and an attempt to implement this computationally would certainly be worthwhile. (Another interesting approach to grouping is found in Tenney and Polansky's work [1980].) Streaming, the sorting of events into contrapuntal lines, is another kind of musical structure that appears to lend itself well to preference-rule modeling. To our knowledge, this has not been attempted, even at an informal level. Finally, key structure seems well suited to a preference-rule approach. Several models have been presented for how key judgments are made for a passage of music. One well-known proposal is Krumhansl's key-profile model (1990), in which the distribution of pitches in a passage is matched to an ideal distribution for each key, with the best match representing the preferred key. One weakness of Krumhansl's model is that it has no mechanism for handling modulation. We are currently investigating some possible solutions to this problem.

Perhaps the most attractive feature of the preference-rule approach, from the point of view of modeling cognition, is the one we first mentioned: it provides a high-level way of describing a cognitive process. In a sense, it is a way of breaking down the problem. The preference-rule system itself can be studied, tested, and refined to determine whether it produces good results without great concern (at least for the moment) for whether the current realization of the system has cognitive validity. As we have said, the search procedure we have proposed has some psychologically attractive features as well, but the preference-rule system itself could still be right even if the search procedure turned out to be completely wrong. Even so, it is still important to have an implementation, since this allows one to test whether one's preference-rule system can really work. Both the successes and failures of our program lend insight into the factors that are important in harmonic and metrical analysis. Combined with other kinds of studies, such as experimental psychological work, we hope that computational studies such as ours will provide converging evidence about the mental structures and processes involved in music cognition.

References

Allen, P., and R. Dannenberg. 1990. "Tracking Musical Beats in Real Time." *Proceedings of the 1990 International Computer Music Conference.* San Francisco: International Computer Music Association, pp. 140–143.

Butler, D. 1992. *The Musician's Guide to Perception and Cognition.* New York: Schirmer.

Chafe, C., B. Mont-Reynaud, and L. Rush. 1982. "Toward an Intelligent Editor of Digital Audio: Recognition of Musical Constructs." *Computer Music Journal* 6(1):30–41.

Cormen, T., C. Leiserson, and R. Rivest. 1990. *Introduction to Algorithms.* Cambridge, Massachusetts: MIT Press.

Desain, P., and H. Honing. 1992. *Music, Mind and Machine.* Amsterdam: Thesis Publishers.

Dowling, J., and D. Harwood. 1986. *Music Cognition.* Orlando, Florida: Academic Press.

Krumhansl, C. 1990. *Cognitive Foundations of Musical Pitch.* Oxford: Oxford University Press.

Lee, C. 1991. "The Perception of Metrical Structure: Experimental Evidence and a Model." In P. Howell, R. West, and I. Cross, eds. *Representing Musical Structure.* London: Academic Press, pp. 59–127.

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

Longuet-Higgins, H. C., and M. Steedman. 1971. "On Interpreting Bach." *Machine Intelligence* 6:221–241.

Maxwell, H. J. 1992. "An Expert System for Harmonic Analysis of Tonal Music." In M. Balaban, K. Ebcioglu, and O. Laske, eds. *Understanding Music with AI.* Cambridge, Massachusetts: MIT Press, pp. 335–353.

Parncutt, R. 1994. "A Perceptual Model of Pulse Salience and Metrical Accent in Musical Rhythms." *Music Perception* 11(4):409–464.

Povel, D.-J., and P. Essens. 1985. "Perception of Temporal Patterns." *Music Perception* 2(4):411–440.

Rosenthal, D. 1992. "Emulation of Human Rhythm Perception." *Computer Music Journal* 16(1):64–76.

Steedman, M. 1977. "The Perception of Musical Rhythm and Metre." *Perception* 6(5):555–570.

Temperley, D. 1997. "An Algorithm for Harmonic Analysis." *Music Perception* 15(1):31–68.

Tenney, J., and L. Polansky. 1980. "Temporal Gestalt Perception in Music." *Journal of Music Theory* 24(2):205–241.

Winograd, T. 1968. "Linguistics and the Computer Analysis of Tonal Harmony." *Journal of Music Theory* 12(1):2–49.