Computational Reconstruction of Cognitive Music Theory

Satoshi Tojo

Japan Advanced Institute of Science and Technology

Keiji Hirata

Future University Hakodate

Masatoshi Hamanaka

University of Tsukuba

tojo@jaist.ac.jp

hirata@fun.ac.jp

hamanaka@iit.tsukuba.ac.jp

Received 15 October 2012

Abstract In order to obtain a computer-tractable model of music, we first discuss what conditions the music theory should satisfy from the various viewpoints of artificial intelligence and/or other computational notions. Then, we look back on the history of cognitive theory of music, i.e., various attempts to represent our mental understandings and to show music structures. Among which, we especially pay attention to the Generative Theory of Tonal Music (GTTM) by Lehrdahl and Jackendoff, as the most promising candidate of cognitive/computational theory of music. We briefly overview the theory as well as its inherent problems, including the ambiguity of its preference rules. By our recent efforts, we have solved this ambiguity problem by assigning parametrized weights, and thus we could implement an automatic tree analyzer. After we introduce the system architecture, we show our application systems.

Keywords Music Information Processing, Cognitive Theory of Music, Computational Musicology, Generative Theory of Tonal Music

§1 Introduction

When we hear music, we sometimes perceive its underlying, deep structure, detached from surface flow of notes. In such cases, we may say we could understand the music piece. If there exists such underlying structure and if we can grasp the meaning of music according to it, can a computer system also retrieve the similar structure? In this section, we discuss the music structure from the three viewpoints: music theory for AI, music as computational object, and music in linguistics. Although all these viewpoints are closely related, each of which possesses its own historical background and still has independent agenda to be discussed.

Music Theory and AI The prime objective of AI is to model our cognition, and to model the real world to give an abstract representation of them. Here, we argue that the representation methods should properly segment the target domain by equivalence relations. In human recognition, however, the equivalence relations do not appear directly; instead, they are perceived as similarity indirectly. According to the MIT Encyclopedia of the Cognitive Sciences,[29] many approaches to modeling similarity can be employed: geometric, featural, alignment-based, and transformational. What we would emphasize here is the fact that every approach to similarity is underlain by the equivalence relations. That is, whatever similarity we think of, it is determined by the extent to which the equivalence relations hold recursively for substructures of music. In other words, we think that a consistent and stable equivalence relation yields a consistent and stable similarity.

In music information research, musical similarity has been drawing attention from many researchers. Some of them are motivated by engineering demands such as music retrieval, classification, and recommendation,[21, 7, 23] and others by modeling the cognitive processes of musical similarity.[5, 6] Now the question is how we can obtain such a similarity, or an appropriate equivalence relation, in the representation of musical objects. Marsden[16] addresses the requirements of a representation system: musical objects must be well-defined and be all grounded to relevant ones in the real world. We think these requirements play an important role in mechanizing music theory. Note that these requirements are almost parallel to formalizing intelligence and representing knowledge.

Since a musical piece contains notes, passages, chords, rhythms, and so on, we can consider various kinds of equivalence relations between them. We show examples of equivalence relations between the two melodies in Fig.1, which shows (a) the incipit of Bach's Invention No.1 and (b) its fake that is transposed a perfect fifth above and notes B are lowered by a semitone. If (a) and (b) are

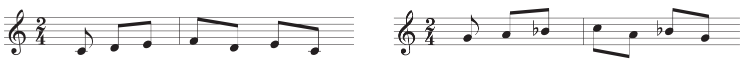


Fig. 1 Three Equivalence Relations of Melodies

compared on the note-wise basis in the literal representation, they are not equal to each other at all. Next, we consider the pitch-interval representations of (a) and (b); for (a) we have $+2, +2, +1, -3, +2, -4$ and (b) $+2, +1, +2, -3, +1, -3$ (unit: semitone). Thus, we find the two elements out of six are identical (the first +2 and the fourth -3). Furthermore, when we employ the Parsons code,[20] where *up* (u) if a note is higher than the previous note, *down* (d) if lower, and *repeat* (r) if the same, then we get u, u, u, d, u, d for both (a) and (b). The difference in resolution among these equivalence relations are determined by an interpretation of musical phenomena or practical requirements.

Here we mention the aspects of commonality and idiosyncrasy in music. In general, the goal of a scientific theory is to understand and represent the mechanism or principle that is common to all phenomena. So in music theory, musicologists have the same attitude to music, but they also seek for the features that make the music unique and peculiar to a composer or an artist. The approach we take to represent music, accessing commonality and idiosyncrasy, influences what and how features of music we should pay attention to.

Music as Computational Object Music has been considered purely emotional, sentimental, and intuitive human affair in general, and thus, has been considered very far from mathematical, formal, procedural operations. The first aim of this article is to introduce a branch of musicology to contradict such a conventional view; i.e., music can be a target of computational operations preserving the gestalt by human cognition. When we say computational operations, we mean rigid input/output relations, that belong to the same domain and also are connected by an algorithm. For example, given a musicXML for a music piece, such algorithms to produce its variation, elaboration, reduction, arrangement, and so on deterministically in the same XML format, can be called computation. In order to achieve this aim, we need to represent music in a symbolically structured way, because every target of computation should be strictly formalized. Therefore, the structure of music must be represented by data structure, which is hierarchical and recursive.[*1] Seemingly, only a limited number of music theories can serve this purpose. If a music piece is represented by a tree then it can also be represented by a recursive feature structure (see Carpenter[2)] and Sag et al.[22)]), which has been well studied and applied to many phases of computation. In some cases, algebraic operations over structures are utilized in a domain; in Section 6.1, we will show examples of *meet* and *join* operations.

Music in Linguistics The analogy between music and natural language has been long discussed. In Bernstein,[1)] the correspondences between a note and a phoneme, a motive and a morpheme, a phrase and a word, and so on, are stated. It is commonly accepted that our human language belongs to a subclass of context-sensitive grammar (CSG) and a superclass of context-free grammar (CFG) in Chomsky hierarchy of formal language; in effect, most sentences can be generated by CFGs, which have long distance dependency and *tree* structure.

The origin of music and language is regarded to be the same (see Wallin et al.[28)]). Actually, we hear natural languages by our ears and utter them by our throats and tongues; i.e., the devices are common. Then, we may consider we recognize music in a similar way to language, employing the same part of our brain. Since a CFG language is accepted by a push-down automaton, and we can employ our short-term memory in our brain as push-down stack, we can assume that music is also governed by a CFG-like grammar. The easiest way to understand how the short-term memory works is the repetition. Since a written scores appeared in the Middle Age in Europe, music has been denoted by an iterated metrical structure by delimiting bars (measures) with equal intervals. Another easy example is melody recognition. In a music piece, the same melody or *phrase* appears repeatedly in time and/or in other voices. The melody/phrase recognition implies that we possess an ability to group consecutive notes together, with the help of short-term memory. Cadence (*Kadenz* in German) is a sophisticated example of long distance dependency, which is a sequence of chords telling us the ending of the music piece. The typical one is

[*1] The signal representation of music is out of scope in this paper.

the progression from V (dominant) to I (tonic).[*2] A music piece begins with a tonic chord of its key in most cases, and we feel a tension when the notes goes far from the original key; on the contrary, when the notes come near to the original tonic chord, we feel a relaxation. Thus far, many natural language researchers have tried to implement music parsers with CFG-like grammar. See Winograd,[30)] Tojo et al,[27)] Steedman,[25)] and so on.

§2 History of Cognitive Music Theory

There have been proposed many music theories[24, 13, 3, 19, 26, 4)] so far. The motivation/objective of each theory is different from each other, and so are the fundamental concepts and models. We will briefly review these in this section.

Schenkerian Analysis Heinrich Schenker (1868–1935) seems the first to present the idea of reduction in music piece. In his reduction analysis, the underlying musical structure resides in three hierarchical levels: the *background*, the *middle-ground*, and the *foreground*. The background possesses the fundamental structure (*Ursatz* in German), and as a result of elaboration we obtain the foreground. The fundamental structure consists of the fundamental line (*Urlinie*), that is the melodic line in the upper voice, and the counterpoint, that is the bass line in the lower voice. Each of melodic/bass lines gives the piece a musical direction. The melodic line proceeds in the descending scale-steps that fill in the interval between the third and the tonic, the fifth and the tonic, or the octave and the tonic. The base voice is a harmonic component of the fundamental structure; it proceeds by an ascending fifth between the tonic and the dominant (I–V), and returns downwards to the tonic (V–I). At the core of Schenkerian analysis, there exists the notion of *prolongation* which means that a pitch event[*3] remains as if it were still sounding. The structure of tonal music is consistent with the structural dominance of the prolonged event. The melodic elaboration from the background to the foreground is called *linear progression*, which horizontalizes the vertical harmony; this process occurs in accordance with the downward scale-steps toward the tonic. On the contrary, in order to preserve the fundamental bass line of I-V-I, the linearization process acts to fill in the intermediate available harmonies. Thus, in the foreground level, horizontalization of the vertical is

[*2] The tonic is the first scale degree of a diatonic scale, and the dominant is the fifth scale degree and is next in importance to the tonic. Here, we represent I and V for the triad chords on the tonic and the dominant.

[*3] A chord or a note, exclusive of a rest.

more emphasized while in the background, the verticalization of the horizontal is salient. The Schenkerian reduction process is better understood in the heuristic terms, rather than in the algorithmical terms, and this fact has alienated the theory from computer implementation.

I-R Model The implication-realization (I-R) theory proposed by Narmour[18] is based on the hypothesis that we listen to the music, always predicting next notes. Narmour assumed that successive three pitch events made fundamental patterns which worked as the gestalt in perception, and the first two produced implication, while the last one might or might not realize the implication. Here, the realization includes continuation, differentiation, and reversal as seen in the gestalt phenomena. The I-R theory can be thought as an alternative to the models based on the reduction of the gestalt. Since a melody can be understood as a series of the triplets of notes, the I-R theory is useful for analyzing melodic contours. Then, the equivalence relations in the I-R theory are that any triplets belonging to an identical pattern cause the same perception and have the same function in the flow of music.

Berklee method Among music theories, the Berklee method is one of the well-known harmony theories. It is quite common in the popular artist community because it is a mechanical and practical rule system so that anyone can develop proper chord progressions. First, the Berklee method developed a nomenclature of chords. Then, if there are two chords having the same name (symbol) yet containing different notes, these two are supposed to work as the same function in a chord progression. Please note that this is an equivalence relation that the Berklee method introduces. As such, the idea that formal grammars should be applied to analyze chord progressions is not new, but it may be inspired by the Chomsky approach to natural languages to some extent. As a result, the Berklee method transformed a chord progression into symbolic manipulation and allows people to generate a progression like playing a puzzle. Although it seems easy, at a first glance, to translate the rules of the Berklee method into a context free grammar, it is not easy to distinguish generic rules of chord progressions from those of exceptional ones. Hence, the rule organization fluctuates, depending on each rule's interpretation. This is mainly because the Berklee method was developed for human, not for a computer.

§3 Generative Theory of Tonal Music

As we have mentioned, if a music piece is represented by a hierarchical tree structure, we can apply various computational operations on it. However, only few theories are eligible for this purpose. We pay attention to the Generative Theory of Tonal Music (GTTM),[13] since it produces a hierarchical tree as a result of analysis of a music piece.

3.1 Overview

GTTM consists of the following four processes. *Grouping analysis* puts group boundaries on a music piece. *Metric Analysis* identifies its rhythmic structure. As a result, the theory composes two kinds of tree structures of music; *Time-span analysis* composes a tree to represent the metric structure and *Prolongational analysis* makes a tree to represent harmonic stability.

Grouping Analysis The goal of this analysis is to find group boundaries. We show several typical rules below, where GPR stands for group preference rules, which will be explained later.

GPR2 (proximity) Consider a sequence of four notes. The transition of the mid two notes may be heard as a group boundary if
a. the interval of time from the end of the 2nd note to the beginning of the 3rd is greater than others.
b. the interval of time between the attack points of the 2nd and the 3rd is greater than others.
GPR3 (change) Consider a sequence of four notes, then the 2nd and the 3rd may be heard as a group boundary, if the followings change.
a. register/ b. dynamics/ c. articulation/ d. length

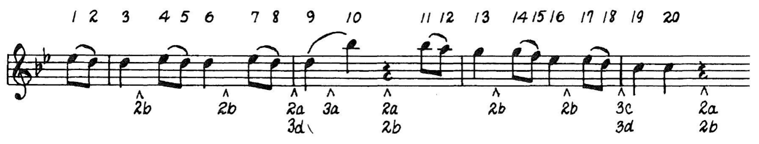


Fig. 2 Boundaries Candidates Shown by Rule Number (Lerdahl and Jackendoff[13, page 47])

We show an example of GPR 2 and 3 in Fig. 2, in which the rule names appear in their applicable places though not all applicable places are marked. Since these rules work disjunctively, the marked places are still candidates of boundaries. As a result of grouping analysis, we obtain a sequence of groups, which do not have gaps nor overlap in each other. These groups are recursively grouped, and compose a hierarchical grouping, as in Fig. 3

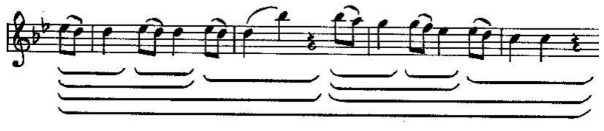


Fig. 3 Hierarchical Grouping in Mozart K.550 (Lerdahl and Jackendoff[13, page 37])

Metric Analysis The metric analysis assigns the beat strength on each pitch event, shown by the number of dots, as in Fig. 4.

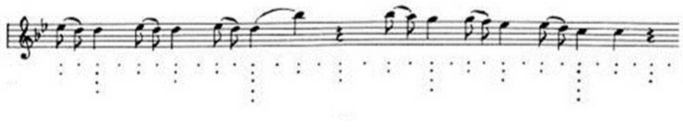


Fig. 4 Metrical Analysis (Lerdahl and Jackendoff[13, page 74])

Time-span Analysis The time-span tree is built by the comparison of adjacent two pitch events in terms of *salience* in the bottom-up way. A pitch event is *salient* when (i) the event is located either at the beginning end or at the terminal end of each group in the grouping analysis (Fig. 3), and (ii) it has a stronger beat in the metrical analysis (Fig. 4). The branch on the more salient pitch event extends upward, absorbing that on the neighboring pitch event. Since the group boundaries are hierarchical, this process continues recursively until one top event is reached. In Fig. 5, the pitch events labelled *a* are more salient than those labelled *b*, and so are those labelled *b* than those labelled *c*. Thus, the pitch events in *b* level and those in *a* level result in reduced music, each of which is shown below *a* (the original) in the figure. In the sequence of reductions, each level should sound like a natural simplification of the previous level. The alternative omission of notes must make the successive levels that sound less like the original. Hence, reduction can be regarded as rewriting an expression to an equivalent simpler one; it often has the same meaning as abstraction. Behind this time-span reduction (step-by-step simplification), there lies the philosophy of the *strong reduction hypothesis*. The listener attempts to organize all the pitch events of a piece into a single coherent structure, in such a way that they are heard in a hierarchy of relative importance.

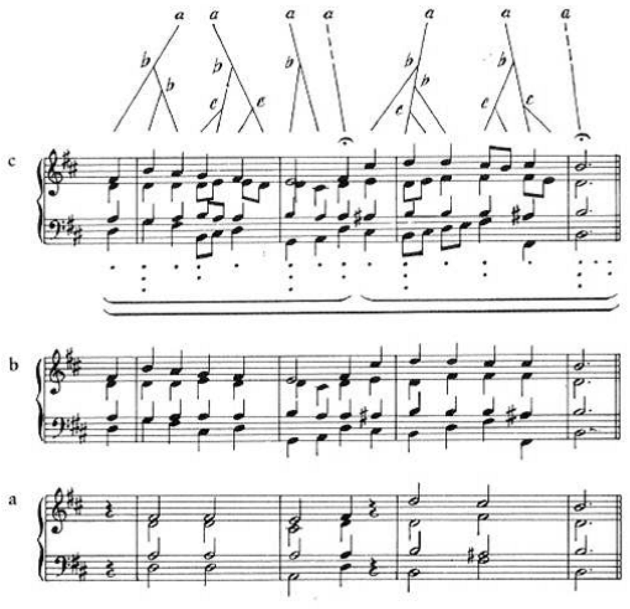


Fig. 5 Time-span reduction in GTTM (Lerdahl and Jackendoff[13, page 132])

Prolongational Analysis While the time-span tree represents the rhythmic stability as it is composed by grouping and metrical analyses, the prolongational tree represents the pitch stability. The notion of *prolongation* means that a pitch event (especially a chord) is still felt remaining and sounding beyond group boundaries, being classified in the following three grades.

- strong prolongation: same bass and root[*4]
- weak prolongation: same root
- progression: no consonant[*5]

Also, motion in music is explained in two ways; we feel *tension* when tones go away from the original tonic, and we feel *relaxation* when they come back to. Tension is represented by right-branching in tree structure, and relaxation

[*4] The root is the note upon which a chord may be built by stacking thirds, while the bass is the lowest note in the chord. When a chord is inverted, the bass becomes different from the root.

[*5] Harmonic notes such as thirds and fifths.

becomes left-branching. These prolongation/progression and tension/relaxation result in six patterns in Fig. 6, where open circle represents strong prolongation, filled-in circle does weak prolongation, and no circle progression.

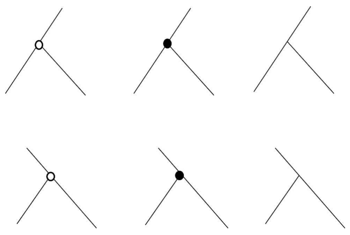


Fig. 6 Six patterns of prolongation/progression

As Schenkerian *Ursatz*, GTTM assumes the existence of *basic form*, which possesses cadential preparation, i.e., the pitch events lead up to the cadence and is represented by doubly embedded left-branching. Including the basic form, a normative form consists of tension (right-branching) in the beginning and relaxation (left-branching) in the cadence, as is shown in Fig. 7.

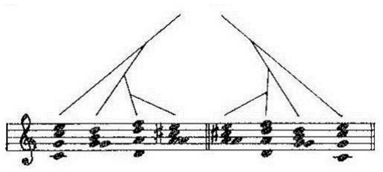


Fig. 7 Normative form in GTTM (Lerdahl and Jackendoff[13, page 191])

The prolongational tree is composed by rearranging the branches of a time-span tree, this time in the top-down way. Choosing the most locally important branch in the time-span tree, the algorithm decides where the branch should be reattached for the prolongational tree from the viewpoint of pitch stability. The stability is stated by the following four conditions.

- Branching condition
  - right strong prolongation > right weak prolongation > right progression
  - left progression > left weak prolongation > left strong prolongation
- Pitch-collection condition: a connection between two events (chords) is more stable if they involve or imply a common diatonic collection.
- Melodic condition
  - Melodically more stable if the distance is smaller.

  - Ascending is most stable in right-branching; descending is most stable in left-branching.
- Harmonic condition
  - Harmonically more stable if their roots are closer on the circle of fifths.
  - Ascending along the circle of fifths is most stable in right-branching; descending along the circle (subdominant to dominant) is most stable in left-branching.

In the left-hand side of Fig. 8, the next important branch after the cadence 'V-I' and the initial 'I', seen from a top-down way, is either the intermediate 'V' or the intermediate 'I' which reaches the next highest in the time-span tree. Since each of 'V' and 'I' can be reattached to either left or right mother branch, there are four possibilities shown by the dotted lines, in the right-hand side of Fig. 8. From the viewpoint of stability, the left-hand side of Fig. 9 is chosen (right strong

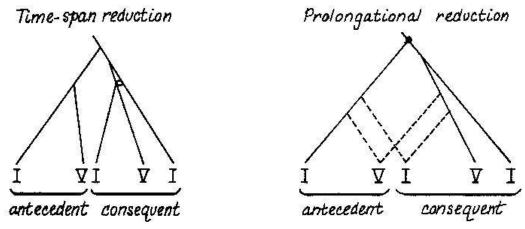


Fig. 8 Four choices in prolongational analysis in GTTM (Lerdahl and Jackendoff[13, page 223])

prolongation), and thus the next important branch on 'V' results in the right branching as in the right-hand side of the figure.

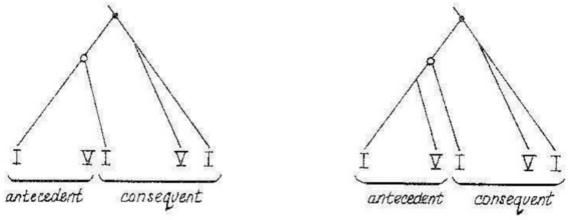


Fig. 9 Ongoing prolongational analysis in GTTM (Lerdahl and Jackendoff[13, page 224])

3.2 Problems of GTTM

Selection of Preference Rules The generative rules of GTTM are classified into well-formedness rules and preference rules. The former rigidly prescribes the grammar; if one of these rules is violated, we cannot compose a tree. The latter states only preferences, and thus the composing procedure becomes indeterministic. In worse cases, those preferences contradict. In Fig. 2, such notations as *2a, 2b, 3a, 3c* and *3d* all show the name of preference rules, each of which claims that the designated place can be a candidate of group boundary. However, if we adopt all these, groups become too minutely minced; thus, we need to give a criterion for each candidate to be true boundaries. We mention how we solve this problem later.

Long Distance Dependency The time-span tree is composed basically in the bottom-up way. This implies that we cannot represent the dependency adequately in some cases. Especially, in the case of dependency between two notes with a long distance, the correspondence should be represented in the top-down manner. In GTTM, such a kind of dependency is explained by an analogy, i.e., 'like a ball thrown and caught' (Lehrdahl and Jackendoff[13], p.133). Fig. 10 shows the theme of the variation of the first movement, piano sonata K.331 in A major, by Mozart. This figure shows the correct time-span tree; the small open circle on the branches at the tail of the piece represents that the sequence of V–I is put together to one event as a cadence.

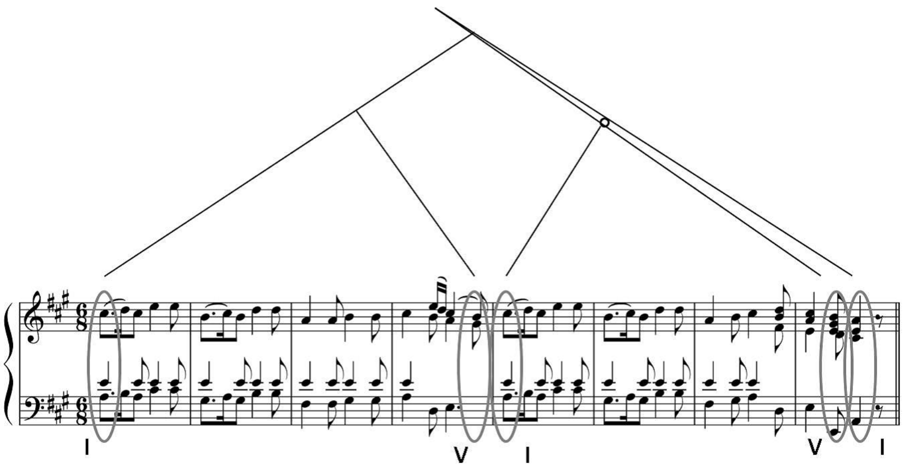


Fig. 10 Cadence in the variation theme, 1st movement of K.331, Mozart

Notice that in the fourth bar (measure), A-E-C♯ at the first beat is I (tonic) in A major and E-G♯-B in the sixth beat is V (dominant); if they are compared, the tonic should be salient (the left-hand side of Fig. 11). However, this V is the nexus to the following I in the next bar, representing half-cadence, and thus this V should be more salient (the right-hand side of Fig. 11) in the macro-scopic view.

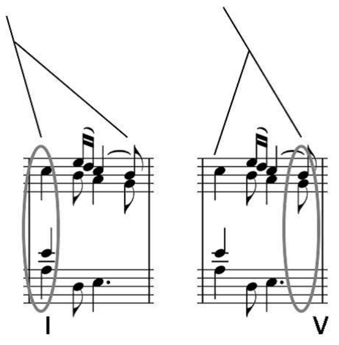


Fig. 11 Saliency in the fourth bar, K331

Thus, we need to equip with a top-down strategy, as well as a bottom-up composition. This issue suggests the following problems of the original theory. First, those multiple preference rules disturb the decision of the order of composition, and result in ambiguity. Second, in order to recognize a correct time-span tree, we need to identify cadence, i.e., we need to find I and V, however, in the original theory the procedure of chord recognition is not mentioned. These chord information is also needed for us to reconstruct prolongational tree, rearranging time-span tree.

§4 Implementing GTTM

In computer implementation of music theory,[?, 3, 19, 26] we have to consider two types of ambiguity in music analysis. One involves human understanding of music, and the other concerns the representation of music theory. The former tolerates our subjective interpretation, while the latter is caused by the incompleteness of the original theory, and the GTTM is not an exception. Therefore, due to the former's ambiguity, we assume there should be more than one correct result. We would like to avoid the latter kind of ambiguity as much as possible. In this section, we extend GTTM by full externalization and parameterization, and propose exGTTM. This externalization in mechanizing GTTM includes introducing an algorithm for generating a time-span tree in the mixed manner of top-down and bottom-up. These parameters control the priorities of rules to avoid conflicts, to obtain the most adequate shape of the hierarchical time-span tree. At this stage, we restrict exGTTM to a monophony, for harmony analysis is not taken into account yet.

4.1 exGTTM – an Introduction

We present the design strategy of a machine-executable extension of GTTM, exGTTM.[8)] Since the original theory lacks formalities in some cases, we need to compensate them for requisite definitions. Thus, we employ full-externalization and parameterization to yield precise controllability and to cover all possible human analyses.

The parameters introduced by exGTTM are classified into the following three categories. For the first category, a parameter is explicitly mentioned in the theory but is not assigned concrete values, hence, we need to valuate such a parameter. For the second category, a parameter appears only implicitly; hence, we need to make it explicit. For example, to resolve the conflict in preference rules, we need to assign a priority value for each preference rule. For the third category, we need to complement several control values which may not possess any musicological meanings, to fully externalize the theory.

The domain of intermediate variables is constrained within the range of 0 to 1, and for this purpose, those variables are normalized at every computing stage. Thanks to this property, exGTTM can flexibly combine any intermediate variables (and possibly parameters) and cascade as many weighted-mean calculations as needed. Accordingly, this facilitates precise controllability.

Among issues that require working algorithms, the problems of acquiring hierarchical structures in the grouping- and metrical-structure analyses and the time-span tree reduction can be all regarded as a constraint-satisfaction problem (CSP). This is because only the expected properties for the hierarchical structures are represented in the form of a rule, that is, no constraint nor order of generating hierarchical structures is determined in advance.

The constraints stipulated by the GTTM rules are divided into two categories: local and global. The former includes GPR2 (proximity, cf. Section 3) and TSRPR1[*6] (strong metrical position), and the latter GPR5[*7] (symmetry)

[*6] Time-Span Reduction Preference Rule 1 – Of the possible choices for a head of time-span T, prefer a choice that is in a relatively strong metrical position.

[*7] Grouping Preference Rule 5 – Prefer grouping analyses that most closely approach the ideal subdivision of groups into two parts of equal length.

and MPR1[*8] (parallelism). We need to handle global constraints carefully when generating hierarchical structures. For the example of GPR5 in Fig.12, given a group at Layer 1, an inner boundary likely occurs around the center of the group, that is, either between Notes 1 and 2 or Notes 2 and 3. Here, we can consider

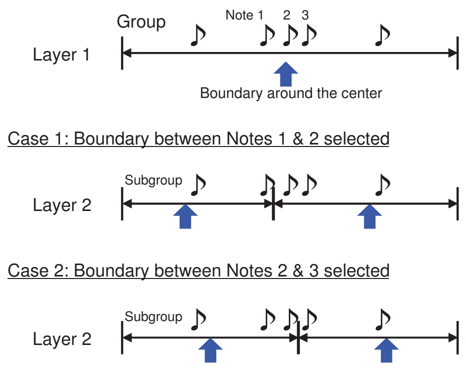

Group
Note 1
2
3
Layer 1
Boundary around the center
Case 1: Boundary between Notes 1 & 2 selected
Subgroup
Layer 2
Case 2: Boundary between Notes 2 & 3 selected
Subgroup
Layer 2

Fig. 12 Global Constraints by GPR5.

two cases as follows. In Case 1, the boundary between Notes 1 and 2 is selected, taking into account the effects of some other rules. Then in each subgroup in Layer 2, the inner boundary of the subgroup may occur in the left-hand side of a center note. On the other hand, in Case 2, the boundary between Notes 2 and 3 is selected. Therefore, the inner boundary may occur in the right-hand side of a center note. Consequently, in computing GPR5, the boundary position always affects the remote boundaries in lower layers, and we have to take into account up-to-date global information every time. That is, the global constraint is inevitably dynamic.

4.2 Overview of exGTTM

Based on the above consideration, we have developed algorithms for generating hierarchical structures for exGTTM so that nodes are generated either from the bottom-most nodes or the top-most node incrementally. In addition, every time the nodes at one layer are calculated, global information is re-calculated before moving onto an adjacent layer.

The exGTTM consists of a grouping structure analyzer, a metrical structure analyzer, and a time-span tree analyzer. Here, we briefly explain the group-

[*8] Metric Preference Rule 1 – Where two or more groups or parts of groups can be construed as parallel, they preferably receive parallel metrical structures.

ing structure analyzer. Fig. 13 is a processing flow of grouping structure analyzer. First, the low-level, local boundaries are found by GPR2, GPR3, and so on. Then, we apply such macro-scopic rules as GPR4 concerning the definition of higher-level boundaries and GPR5 which stipulates that the length of groups should be well-balanced. The latter process is repeated recursively as in the figure.

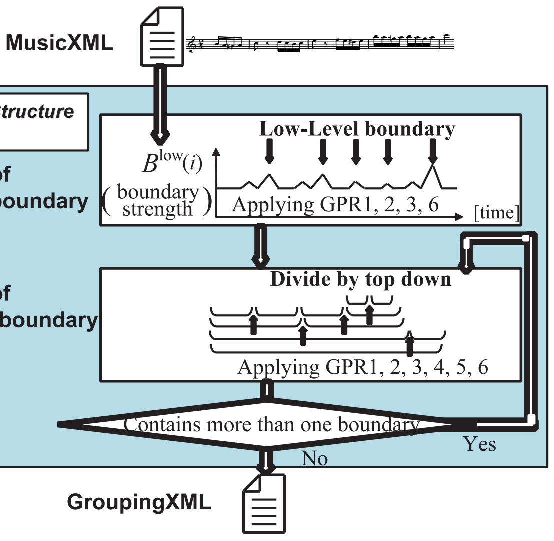

MusicXML
Grouping-Structure Analyzer
Detection of low-level boundary
Low-Level boundary
$B^{\text{low}}(i)$
(boundary strength)
Applying GPR1, 2, 3, 6
[time]
Detection of high-level boundary
Divide by top down
Applying GPR1, 2, 3, 4, 5, 6
Contains more than one boundary
Yes
No
GroupingXML

Fig. 13 Processing flow of grouping structure analyzer.

Here we introduce two kinds of parameters.

$D_r(i)$ means the degree as to how strongly the rule holds, and
S_r means the weight of the rule,

where suffix r is the rule number and argument i means an ordinal number of a note. In Fig. 14, $B^{\text{low}}(i)$ represents the local strength by a real number between 0 and 1; the larger the value is, the more likely the boundary is.

$$B^{\text{low}}(i) = \sum_R D_{\text{GPR}_R}(i) \times S_{\text{GPR}_R} / \max_{i'} \left(\sum_R D_{\text{GPR}_R}(i') \times S_{\text{GPR}_R} \right)$$

for $R \in \{2\text{a}, 2\text{b}, 3\text{a}, 3\text{b}, 3\text{c}, 3\text{d}, \textit{and}\ 6\}$. $B^{\text{high}}(i)$ represents the strength of a boundary in a higher hierarchy by a real number between 0 and 1. $B^{\text{high}}(i)$ is different from $B^{\text{low}}(i)$ in that the former reflects the result of those rules that

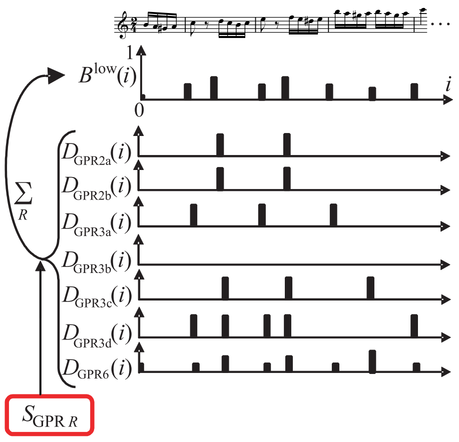


Fig. 14 Low-level strength of boundary $B^{\mathrm{low}}(i)$.

concern phrasal structure, that is, $D_{\mathrm{GPR4}}(i)$ and $D_{\mathrm{GPR5}}(i)$. When a group includes a local boundary in it, the boundaries of one upper level $\hat{i}$ is recursively detected by the following procedure (Fig. 15), where $B^{\mathrm{high}}(i)$ is renewed at each level, since the value of $D_{\mathrm{GPR5}}(i)$ will change at every grouping level. Then we have a boundary

$$\hat{i} = \underset{i}{argmax}\, D^{\mathrm{high}}(i),$$

where

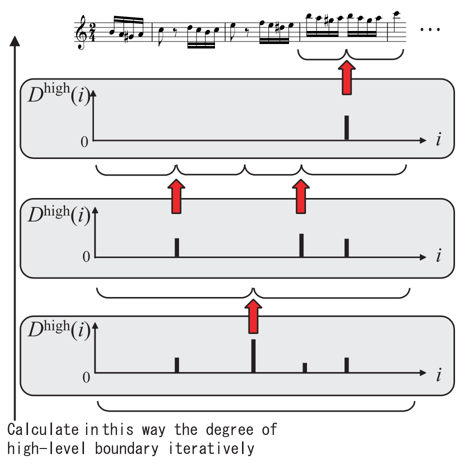
$$D^{\mathrm{high}}(i) = D^{\mathrm{low}}(i) \times B^{\mathrm{high}}(i),$$

$$B^{\mathrm{high}}(i) = \sum_R D_{\mathrm{GPR}_R}(i) \times S_{\mathrm{GPR}_R} / \max_{i'} \left(\sum_R D_{\mathrm{GPR}_R}(i') \times S_{\mathrm{GPR}_R} \right)$$

for $R \in \{2\mathrm{a}, 2\mathrm{b}, 3\mathrm{a}, 3\mathrm{b}, 3\mathrm{c}, 3\mathrm{d}, 4, 5, \mathit{and}\ 6\}$ and for all the i's included in the group.

§5 ATTA and Interactive GTTM Analyzer

We have implemented exGTTM on a computer, and have named ATTA: *Automatic Time-span Tree Analyzer*. The grouping and metrical analyses, and the resulting time-span tree structures may change depending on the adjustable parameters. In the current stage, the parameters are configured by humans because the optimal values of the parameters depend on the piece of music.



$D^{high}(i)$
0
i
Calculate in this way the degree of high-level boundary iteratively

Fig. 15 Construction of hierarchical grouping structure.

5.1 Analyses by ATTA

Mozart Sonata K. 331 can be interpreted in two different ways: as a grouping structure that has a boundary between note 4 and note 5 (Fig. 16(a)) and as a grouping structure that has a boundary between note 5 and note 6 (Fig. 16(b)). The system can output both grouping structures properly as a result of using exGTTM for analysis by configuring S_{GPR2a}, S_{GPR2b}, and S_{GPR3a}.

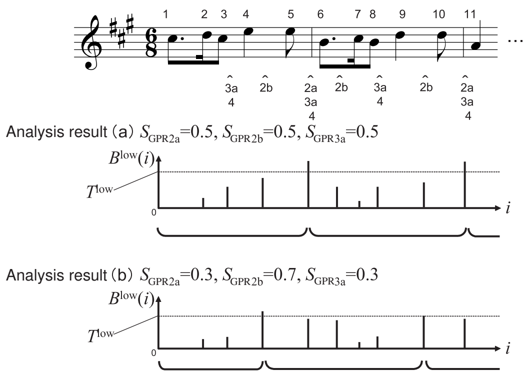


Fig. 16 Analysis of Mozart Sonata K. 331.

Fig. 17 shows the analyses of two pieces, which are tuned with the same parameters, i.e., Beethoven, Turkish March and English Traditional, Green Sleeves. The numbers at the separations of the tree in Fig. 17 indicate the applicable rules. As a result of comparing two pieces, the same rules hold, i.e., TSRPR1 and TSRPR3b.[*9]

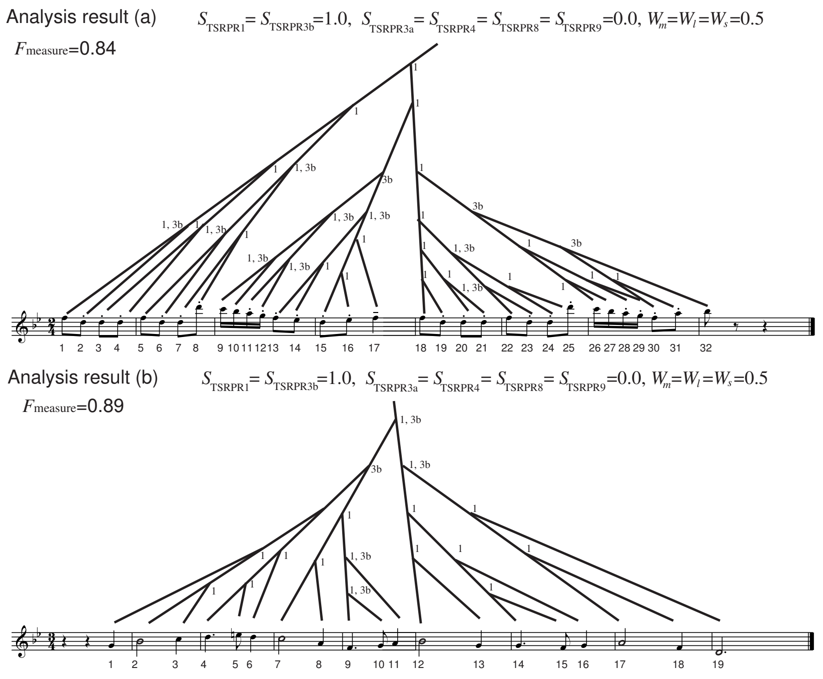

Analysis result (a)
$S_{\mathrm{TSRPR1}} = S_{\mathrm{TSRPR3b}} = 1.0$, $S_{\mathrm{TSRPR3a}} = S_{\mathrm{TSRPR4}} = S_{\mathrm{TSRPR8}} = S_{\mathrm{TSRPR9}} = 0.0$, $W_m = W_l = W_s = 0.5$
$F_{\mathrm{measure}} = 0.84$
Analysis result (b)
$S_{\mathrm{TSRPR1}} = S_{\mathrm{TSRPR3b}} = 1.0$, $S_{\mathrm{TSRPR3a}} = S_{\mathrm{TSRPR4}} = S_{\mathrm{TSRPR8}} = S_{\mathrm{TSRPR9}} = 0.0$, $W_m = W_l = W_s = 0.5$
$F_{\mathrm{measure}} = 0.89$

Fig. 17 Analysis of two pieces having the same parameter sets. (a) Beethoven, Turkish March. (b) English Traditional, Green Sleeves.

5.2 Interactive GTTM analyzer

Fig. 18 is a screenshot of the viewer of our interactive GTTM analyzer. There is a sequence of notes displayed in a piano roll format. Fig. 19 is an overview of our interactive GTTM analyzer, consisting of the ATTA, a manual editor, and a process editor. As the ATTA cannot output all analyses that correspond to all the interpretations of a piece of music, we designed a manual editor, which generates all analyses. The original theory includes feed-back operations from higher- to lower-level in the tree structure; however, no detailed description is given but only a few examples are given. To solve this problem, we developed a process editor, which enables seamless adjustment of the automatic analy-

[*9] Time-Span Reduction Preference Rule 3b – Of the possible choices for head of a time-span T, weakly prefer a choice that has a lower bass pitch.

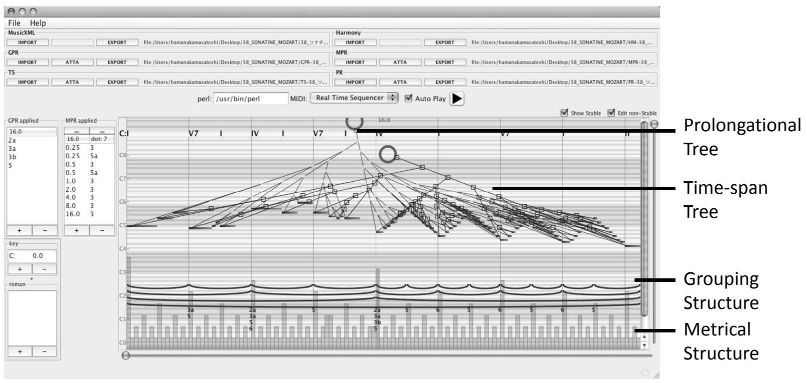


Fig. 18 Screenshot of interactive GTTM analyzer.

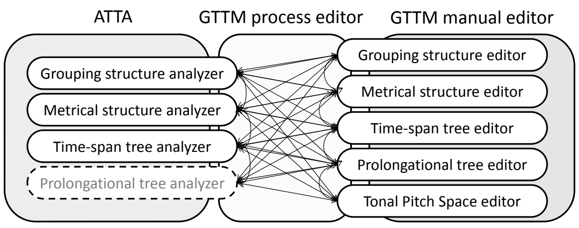

ATTA
GTTM process editor
GTTM manual editor
Grouping structure analyzer
Metrical structure analyzer
Time-span tree analyzer
Prolongational tree analyzer
Grouping structure editor
Metrical structure editor
Time-span tree editor
Prolongational tree editor
Tonal Pitch Space editor

Fig. 19 Overview of interactive GTTM analyzer.

sis process with an ATTA and the manual edit process with a manual editor. Therefore, a user can acquire the target analysis by iterating the automatic and manual processes interactively and can easily reflect his or her interpretations on a piece of music.[*10] We also publicize the database of three hundred pairs of scores and correct data which we asked musicologists to manually analyze the score data faithfully with regard to the theory using our interactive GTTM analyzer.

The interactive GTTM analyzer is the first application for acquiring time-span trees. We hope to benchmark the analyzer to other systems, which will be constructed. We use the XML as the import and export format since XML format is extremely convenient to express hierarchical musical structures.

§6 Applications

This section we describe two application systems called *ShakeGuitar*

[*10] We publicize our interactive GTTM analyzer at:
`http://music.iit.tsukuba.ac.jp/hamanaka/gttm.htm`

and *Expectation Piano*, both of which employ ATTA and the interactive GTTM analyzer.

6.1 ShakeGuitar

ShakeGuitar is a demonstration system for the melody morphing method that changes the morphing level of each half bar by using the values from the accelerometer in the iPhone / iPod Touch/ iPad (Fig. 20).[10] When the user stops moving the iPhone / iPod Touch/ iPad, the unit plays the backing melody of *The Other Day, I Met a Bear (The Bear Song).* When the user shakes it vigorously, it plays heavy soloing. When the user shakes it slowly, it plays a morphed melody between the backing and the heavy soloing.

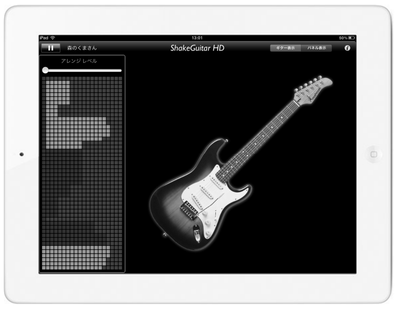


Fig. 20 ShakeGuitar.

In melody morphing, we use primitive operations of the subsumption relation (written as $\sqsubseteq$ in Fig. 21(a)), meet (written as $\sqcap$) and join (written as $\sqcup$). The meet operator extracts the largest common part or the most common information of the time-span trees of two melodies in a top-down manner (Fig. 21(b)). The join operator unites two time-span trees in a top-down manner as long as the structures of two time-span trees are consistent (Fig. 21(c)).

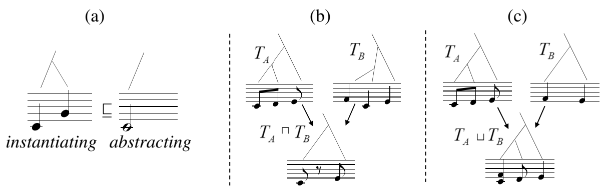


Fig. 21 Examples of Subsumption, meet and join.

The meet operation $T_A \sqcap T_B$ is an abstraction from T_A and T_B, and as a result such notes as only in T_A or only in T_B are discarded (Fig. 22(1)). Since we would like to retrieve those lost notes gradually, we find an adequate subreduction, called *Melody division reduction*, on each reduction path (Fig. 22(2)). Finally, we employ the join operator to combine melodies C and D, residing on reduction paths, to obtain a morphing (Fig. 22(3)).[10]

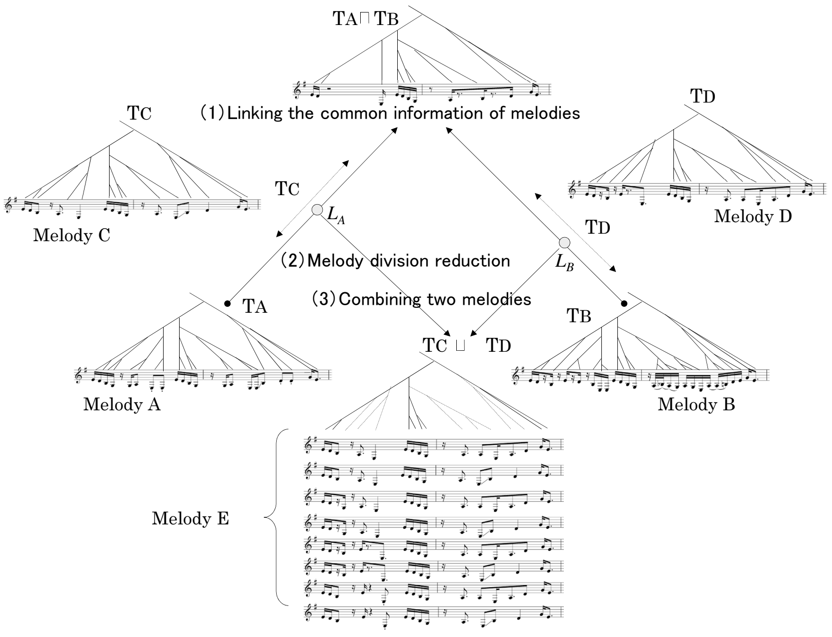

TA ⊓ TB
(1) Linking the common information of melodies
TC
TD
Melody C
Melody D
TC
L_A
TD
L_B
(2) Melody division reduction
(3) Combining two melodies
TA
TB
TC ⊔ TD
Melody A
Melody B
Melody E

Fig. 22 Overview of melody morphing method.

6.2 Expectation Piano

Our expectation piano assists novices with musical improvisation by displaying the predicted notes on the piano lid (Fig. 23).[11] When the novice finds it hard to continue playing the melody, she/he can continue the improvisation by playing a note displayed on the lid, without impairing tonality.

The predicted notes are displayed in piano roll format within the range of view of the keyboard. The roll scrolls down at a constant speed. Below the piano lid, which is made of semitransparent acrylic resin, there is a 32 × 25 full-color LED matrix for displaying the scrolling piano roll. 32 represents two measures when the resolution is a sixteenth note, and 25 is the number of keys on the keyboard. The color of each LED in the matrix is determined under

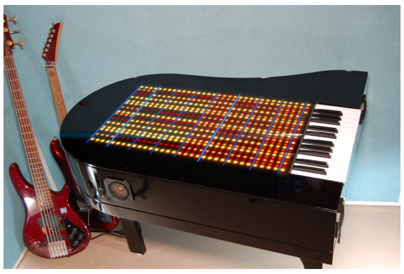


Fig. 23 Expectation piano.

the assumption that the onset of the next note will start at the corresponding position on the piano roll and by calculating the level of stability. When the level of stability is high, the LEDs show yellow, when it is low, they show black, and when it is neither, they show red. There is also a 32 × 20 blue LED matrix that displays the bar lines of the piano roll.

We identified two key requirements for our melody expectation method to make it useful for musical novices to play an improvisation on the expectation piano. 1) Candidate notes are predicted and output even if the input melody is novel. 2) The output is appropriate from a musical point of view. Two approaches were considered when developing this method: statistical learning and music theory. With the statistical learning approach, the predictions depend on the characteristics of the data used for learning: composer, genre, period, country, etc.[17, 15] Moreover, predicting candidate notes for a novel melody is problematic because the system may not be able to find a similar melody in the learning data and thus, may be unable to evaluate whether the notes are appropriate or not. With the music theory approach, the predictions do not depend on the characteristics of the data used for learning. It can thus be applied to novel melodies.

Our melody expectation method predicts candidate notes by using the level of stability of the time-span tree.[9] The main advantage of our melody expectation method is that, the stability of a melody is calculated by analyzing the whole melody from the beginning note to the expected note, not from only the local melody (a few notes previous to a relevant note); previous melody expectation methods based on music theory (e.g. Steve Larson's theory of musical forces[12]) have derived the expected note from the local melody. Music tends to be more interesting when it does not match with the listener's expectation, such as a delayed note, and this may result in tension and relaxation. A composer can deliberately construct such music, which can make it difficult to predict the next notes in the melody with accuracy. For example, an ornamental note is often inserted before the expected note. In such cases, our method can predict candidate notes fairly well because it can evaluate the stability of the entire structure of the time-span trees, which includes branches connected to essential notes and leaves connected to ornament notes.

The aim of Expectation Piano is to assist novices when they cannot choose adequate notes for natural progression. Here, the level of stability should represent the performer's intuition. Thus, we investigate how the level fluctuates in music pieces. Fig. 24 shows the results for Haydn's andante. The graph below the musical staff indicates the level of melody stability, corresponding to each above note. The number under each note indicates the level of stability for selecting a pitch of 25 possible ones. Stability of a pitch event is calculated from the partial melody between its beginning and the relevant event, but not by the sole event. As a result, the F# in measure 7 does not have the lowest stability, and that of C in measure 5 is lower than that of G in the previous measure, even in C major. Although this may contradict intuition, the calculated result is faithful to our algorithm. At the end of the 4th and 8th measures, the level of stability is high. This is because a dominant chord's note that wants to resolve to a tonic chord's note occurs. In contrast, at the beginning of the 5th measure, the level of stability is relatively low. This is because a tonic chord's root note at the beginning of the 5th measure occurs, and various progressions can follow the root note. These results show that our prediction method works well from a musical point of view.

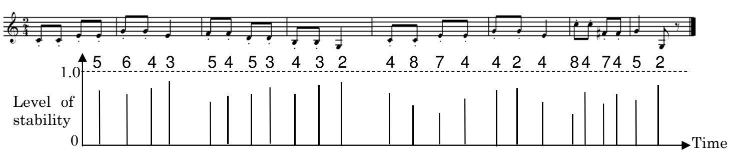


Fig. 24 Example of melody expectation.

§7 Summary

We argue that we need to rely on a solid music theory to yield a consistent and stable model, in which proper equivalence relations hold between represented musical objects. As we have mentioned in Chapter 1, these representations of notes and other musical structures must be properly grounded onto the real world. Furthermore, to embody a music theory into a computer system, we must overcome some widely recognized intrinsic difficulties. One is to give firm formalization to those ambiguous concepts, and the other is to complement some missing concepts. We have chosen the Generative Theory of Tonal Music (GTTM) by Lehrdahl and Jackendoff as the most solid theory of music. We have solved those intrinsic difficulties, by assigning parametrized weights to the preference rules, and have fully externalized the theory into exGTTM, together with the automatic tree analyzer ATTA. We have introduced two of our application systems, *ShakeGuitar* and *Expectation Piano*, and thus could show that the theory is utilized as a basis of computational musicology.

References

1) Bernstein, L., The Unanswered Question - Six Talks at Harvard by Leonard Bernstein, Kulture in DVD, 1976.

2) Carpenter, B., *The Logic of Typed Feature Structures*, Cambridge University Press, 1992.

3) Cooper, G. and Meyer, L. B., *The Rhythmic Structure of Music*, The University of Chicago Press, Chicago, 1960.

4) Cope, D., Computer Models of Musical Creativity. The MIT Press, 2005.

5) ESCOM: 2007 Discussion Forum 4A. Similarity Perception in Listening to Music. MusicæScientiæ, 2007.

6) ESCOM: 2009 Discussion Froum 4B. Musical Similarity. Musicae Scientiæ, 2009.

7) Grachten, M., Arcos, J.-L., de Mantaras, R.L., *Melody retrieval using the Implication/Realization model*, MIREX, 2005.

8) Hamanaka, M., Hirata, K. and Tojo, S., Implementing "A Generating Theory of Tonal Music", *Journal of New Music Research*, Vol. 35, No. 4, pp. 249-277, 2007.

9) Hamanaka, M., Hirata, K. and Tojo, S., FATTA: Full Automatic Time-span Tree Analyzer, *Proceedings of the 2007 International Computer Music conference(ICMC2007)* , Vol. 1, pp. 153-156, 2007.

10) Hamanaka, M., Hirata, K. and Tojo, S., Melody Morphing Method based on GTTM, *Proceedings of the 2008 International Computer Music conference(ICMC2008)*, pp. 155-158, 2008.

11) Hamanaka, M., Hirata, K. and Tojo, S., Melody Expectation Method Based on GTTM and TPS, *Proceedings of the 9th International Conference on Music Information Retrieval conference(ISMIR2008)* , pp. 107-112, 2008.

12) Larson, S., Musical Forces and Melodic Expectations: Comparing Computer Models with Experimental Results. *Music Perception* , 21/4, 457-498, 2004.

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

14) Lerdahl, F., *Tonal Pitch Space*, Oxford University Press, 2001.

15) Lo, M. Y. and Lucas, S.M., Evolving Musical Sequences with N-Gram Based Trainable Fitness Functions, *Proceedings of the 2006 IEEE Congress on Evolutionary Computation* , pp. 601-608, 2006.

16) Marsden, A., Generative Structural Representation of Tonal Music. *Journal of New Music Research* Vol.34, No.4, pp.409-p428, 2005.

17) Mozer, M., Neural Network Music Composition by Prediction: Exploring the Benefits of Psychoacoustic Constraints and Multi-Scale Processing. *Connection-Science*, Vol. 6, No. 2-3, pp. 247-280, 1994.

18) Narmour, E., The Analysis and Cognition of Basic Melodic Structures – The Implication-Realization Model, Univ. of Chicago Press, 1990.

19) Narmour, E., *The Analysis and Cognition of Basic Melodic Structure*, The University of Chicago Press, Chicago, 1990.

20) Parsons, D., The Directory of Classical Themes, Piatkus Books, 2008. `http://en.wikipedia.org/wiki/Parsons_code`

21) Pampalk, E., *Computational Models of Music Similarity and their Application in Music Information Retrieval*,PhD Thesis, Vienna University of Technology, 2006.

22) Sag, I. A. and Wasow, T., *Syntactic Theory – A Formal Introduction*, CSLI Publications, 1999.

23) Schedl, M., Knees, P., Böck, S., Investigating the Similarity Space of Music Artists on the Micro-Blogosphere. In *Proceedings of ISMIR 2011*, pp.323–328, 2011

24) Schenker, H. (Oster, E. (trans.)) Free Composition, Longman, 1979. Original: Der Freie Satz, 1935.

25) Steedman, M., The Blues and the Abstract Truth: Music and Mental Models. In A. Garnham and J. Oakhill, (eds.), Mental Models In Cognitive Science. pp.305-318. Mahwah, NJ: Erlbaum, 1996.

26) Temperley, D., *The Cognition of Basic Musical Structures*, The MIT Press, Cambridge, 2001.

27) Tojo, S., Oka, Y., and Nishida, M., Analysis of Chord Progression by HPSG. in *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications*, 2006

28) Wallin, N., Merker, B., and Brown, S. (ed.) *The Origins of Music*, The MIT Press, 2000.

29) Wilson, R.A., Keil, F. (Eds). *The MIT Encyclopedia of the Cognitive Sciences*, The MIT Press, 1999.

30) Winograd, T., Linguistics and the computer analysis of tonal harmony, In *Journal of Music Theory*, vol.12, no.1, 1968.