Applying Melody-Morphing Method to Composition

Masatoshi Hamanaka[1], Keiji Hirata[2] and Satoshi Tojo[3],

[1] RIKEN
[2] Future University Hakodate
[3] JAIST
masatoshi.hamanaka@riken.jp

Abstract. We describe the application of a melody-morphing method based on the generative theory of tonal music to composition. The melody morphing method uses mathematical operations to generate an intermediate melody between one melody and another. Although we have been evaluating the basic performance of the method, we have not previously used it for music production. Therefore, we conducted an experiment with a musicologist who understood the melody morphing method and attempted to use it for composition. When a melody generated with the method was unnatural, the musicologist added notes to correct it. The experimental results showed that 78.5% of the notes in the composed pieces were generated with the melody morphing method, while the remaining 21.5% was added by the musicologist.

Keywords: generative theory of tonal music (GTTM), melody morphing method, time-span tree, composition.

1 Introduction

For over 10 years, we have been studying musical operations based on the generative theory of tonal music (GTTM) [1-4]. We believe that, by using the time-span trees obtained from GTTM analysis, it is possible to express musical arranging processes as operations like in mathematics. If it is possible to express such a process performed by a composer in terms of operations, then the process can be defined in the form of operations and reused. As the adjustment of a piece requires a large amount of simple repetitive work, our goal is to reduce this work by reusing operations.

The melody morphing method that we previously developed is one example of expressing an adjustment process by operations [5, 6]. We also developed a demonstration application for the melody morphing method, called "ShakeGuitar." We have never used this method for composition, however, except for sound source creation for ShakeGuitar.

We conducted an experiment to verify the effectiveness of the melody morphing method. In the experiment, a musicologist who understood the method used it to manually generate an intermediate melody. If the intermediate melody was unnatural, the musicologist added notes to correct it. We found that 78.5% of the notes were generated with the melody morphing method, while the remaining 21.5% was manually added by the musicologist.

2 Related Work

Many melody-generation methods have been proposed. For example, Papadopoulos et al. used high-order Markov models to generate a melody from fragments of an existing composition [7]. Herremans and Sörensen used a variable neighborhood search for melody generation and provided a plug-in for open-source music composition and a notation program called MuseScore [8]. Boulanger-Lewandowski et al. used the Recurrent Temporal Restrictive Boltzmann Machine to generate polyphonic pieces [9]. Roig et al. applied a probabilistic model to develop a model for generating music [10]. Bemman and Meredith constructed an algorithm that can generate specific all-partition arrays used in three of Milton Babbitt's works [11]. Herremans and Chew used an efficient variable neighborhood search optimization algorithm to generate polyphonic music with recurring themes and tension profiles [12].

The advantage of morphing is not only its simple and accurate transferability and manipulability but also the ease of understanding the relationship between inputs and outputs. Suppose that a music-generation system can accept morphing commands, enabling the user to issue a simple command, such as "add the nuance of melody B to melody A," to accurately convey the user's intention to the morphing system. As a result, the system would generate multiple melodies representing intermediate steps from melody A to melody B.

We composed artificial variations of Mozart's Twelve Variations on "Ah vous dirai-je, Maman", K. 265/300e from existing variations by using our melody morphing method. We evaluated these synthesized variations according to the impression of human listeners and found an interesting correspondence between the theoretical distance and psychological distance [13]. In this study, we evaluated the method through manual operation because the method requires manual operation, as discussed in Section 5, to generate the morphed melody.

3 Time-Span Tree of GTTM

Melody morphing uses time-span trees obtained from the results of GTTM analysis. The GTTM consists of four modules, each of which assigns a separate structural description to a listener's understanding of a piece of music. As shown in Fig. 1, the four modules output a grouping structure, metrical structure, time-span tree, and prolongational tree. The time-span tree is a binary tree, which is a hierarchical structure, representing the relative structural importance of notes that differentiate the essential parts of a melody from ornamentation.

3.1 Abstraction of melody

Figure 2 shows an example of abstracting a melody by using a time-span tree. The figure includes a time-span tree from melody, D, which embodies the results of GTTM analysis. In the time-span tree, important notes are connected to branches

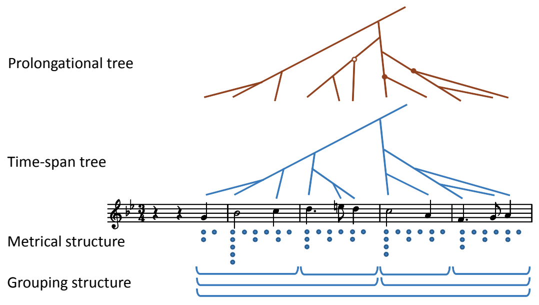

Fig. 1. Prolongational tree, time-span tree, metrical structure, and grouping structure obtained from GTTM analysis

nearer the root of the tree, while unimportant notes are connected to leaves. We can obtain an abstracted melody, E, by slicing the tree in the middle (line E) then omitting notes whose branch connections are below line E. In the same manner, if we slice the tree higher up at line F, we can obtain an even more abstracted melody, F. We can regard this abstraction of melody as a kind of melody morphing because melody E is an intermediate melody between melodies D and F.

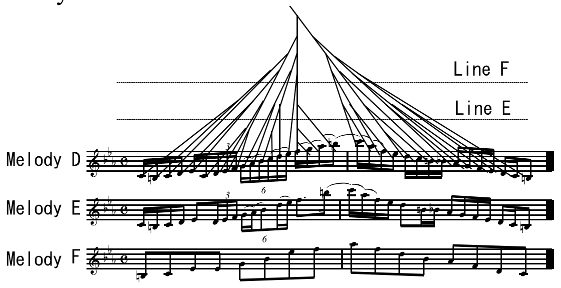

Fig. 2 Abstraction of melody

3.2 Inverse process of composition

The analysis of a time-span tree is effectively an inverse process of composition. In this subsection, we explain such time-span tree analysis in terms of the composing process of Mozart for his Piano Sonata No. 11 in A major, K. 331/300i. Note that the following description follows one possible interpretation of K. 331/300i, but other interpretations also exist.

Figure 3a shows a simple chord progression in K. 331/300i, which repeats the first and fifth chords. In other words, K. 331/300i has a basic pattern of tonic, dominant, tonic, and dominant. The time-span tree for this pattern is shown in blue in Fig. 3. Figure 3b represents the composer's first changes. In the first and second bars, he added position change notes, meaning notes with a difference of one octave. The position change notes are connected to the time-span tree as child nodes, shown in yellow. Furthermore, in the fourth bar the composer divided the fifth chord into I-V, so the I chord is connected to the V chord as a child.

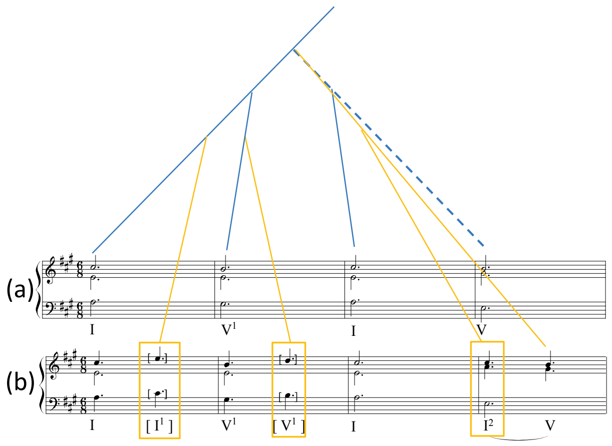

Fig. 3 First step in composing K. 331/300i

Figure 4c shows the next composing step from the melody in Fig. 4b (previously shown in Fig. 3b). The first and second bars incorporate motifs three semitones up. In the third bar, the composer wants to repeat the same motif. The note he uses is E rather than D because the use of D would violate the prohibition of parallel fifths. The added notes become a child of the previous notes in the bar.

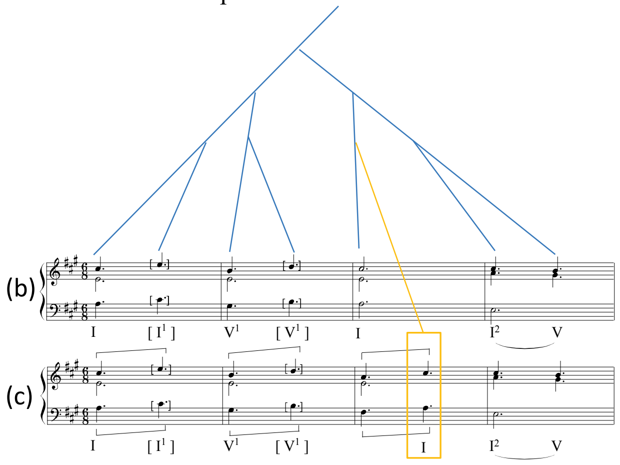

Fig. 4 Second step of composing K. 331

Next, in the first and second bars of Fig. 5d, the composer added auxiliary notes, and in the third bar, he added passing notes. The auxiliary and passing notes become child branches of the branches for the first beat in each bar. Furthermore, in the fourth bar and last beat of the third bar, the composer again divided the I-V progression into II-I-V. The II chord is connected to the I chord in the third bar (i.e., the previous chord) because a branch of a time-span tree cannot cross a strong beat. In other words, as shown with the red "X" in Fig. 5c, the II chord cannot be connected to the I-V chords in the fourth bar, even the II was divided from the I -V.

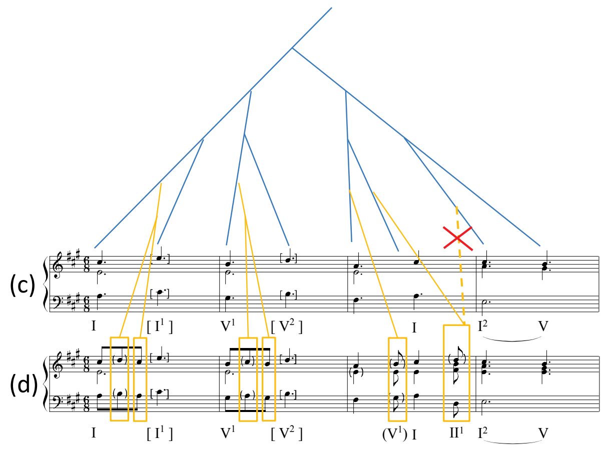

Fig. 5 Third step in composing K. 331/300i

Then, in the first and second bars of Fig. 6e, the composer changed the rhythm and added two notes at the end of each bar. The added notes are connected to the chords of the previous beats. Moreover, in the third and fourth bars, the composer stretched a half bar to a full bar and compressed one and a half bars into one bar. The I-II chords are then connected to the I-V chord because there is a strong beat before the I-II chords, and a branch cannot cross this beat.

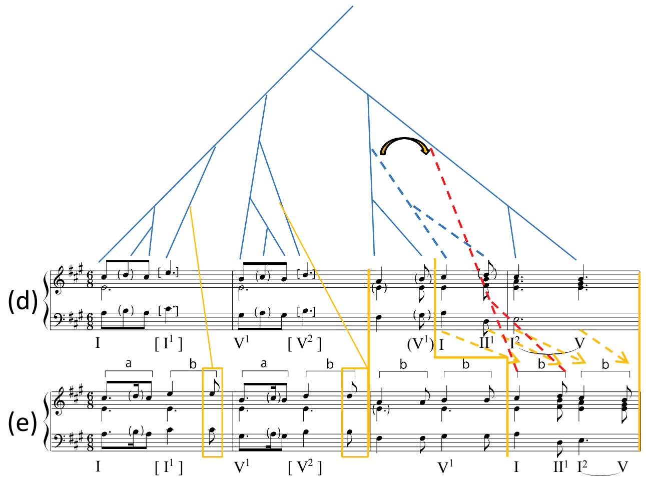

Fig. 6 Fourth step in composing K. 331/300i

Finally, in the last bar of Fig. 7f, the composer added a changing note before an imperfect cadence to highlight the imperfect cadence. The changing note is connected as a child.

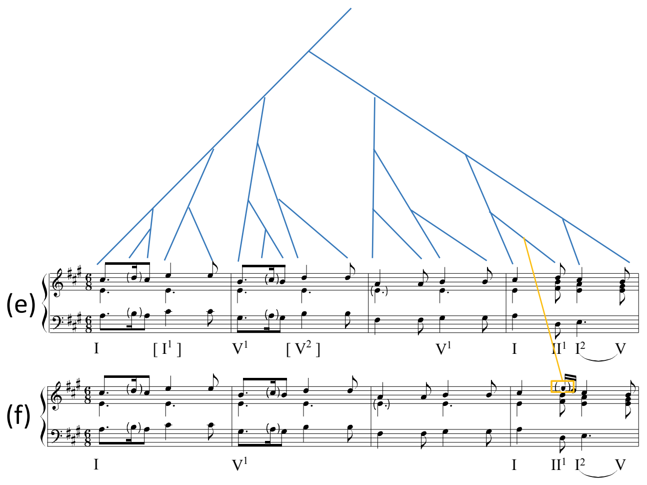

Fig. 7 Final step in composing K. 331/300i

Figure 8 shows the resulting time-span tree of K. 331/300i. As described in Section 3.1, we can obtain an abstracted melody by slicing the tree in the middle and omitting notes whose branch connections are below the line. Therefore, time-span reduction is the inverse process of composition.

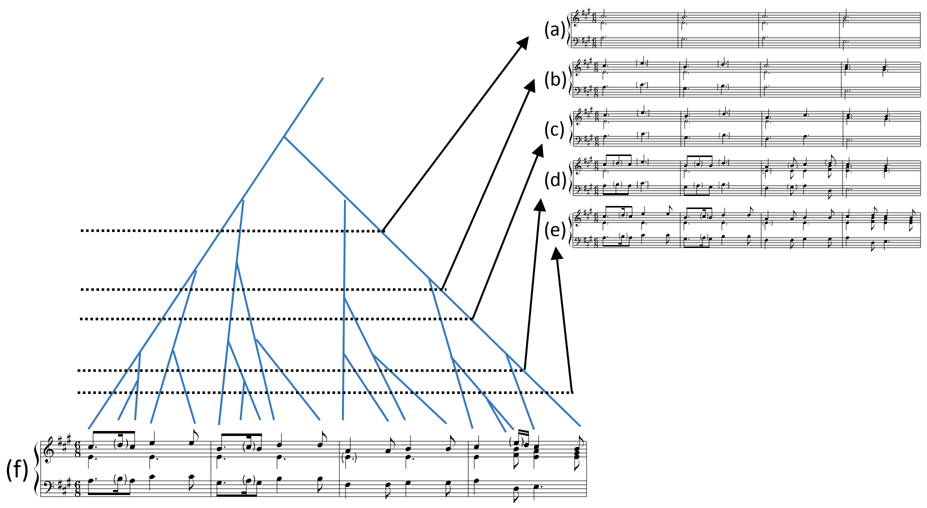

Fig. 8 Time-span tree of K. 331/300i and reduced melodies

3.3 Primitive operations of time-span trees

To implement melody morphing, we use the primitive operations proposed by Hirata [14]: the subsumption relation (written as ⊑), meet (written as ⊓), and join (written as

$\sqcup$). As shown in Fig. 9a, subsumption represents the relation by which "an instantiated object subsumes an abstract object." For example, the relationship among T_D, T_E, and T_F, which are the time-span trees (or reduced time-span trees) of melodies D, E, and F in Figure 2, can be represented as follows:

$$T_F \sqsubseteq T_E \sqsubseteq T_D \quad (\mathbf{1})$$

Figure 9b illustrates the meet operator, which extracts the largest common part or most common information of the time-span trees of two melodies in a top-down manner. Finally, Fig. 9c illustrates the join operator, which joins two time-span trees in a top-down manner as long as their structures are consistent.

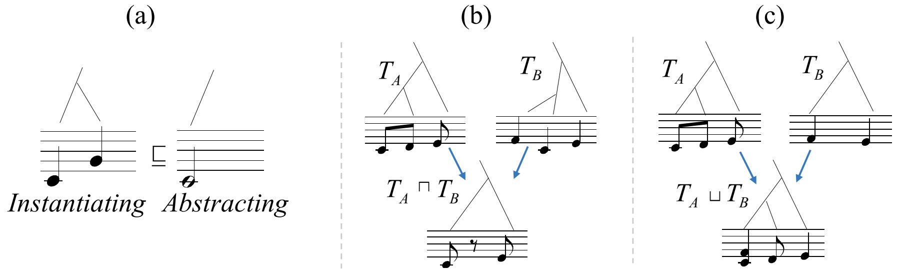

Fig. 9 Examples of subsumption $\sqsubseteq$, meet $\sqcap$, and join $\sqcup$ operations

4 Melody Morphing Method Based on GTTM

In this section, we explain the melody morphing method based on GTTM we proposed in 2008 [5]. The initial melody A, target nuance melody B, morphing result melody C, and melody morphing method must meet the following conditions: 1 and 2 for melody C, and 3 and 4 for the method.

1. The similarity of A and C must be greater than that of A and B, and the similarity of B and C must also be greater than that of A and B.
2. When B is the same as A, C will be the same as A.
3. The output of multiple Cs depends on parameters that determine the levels of influence of the features of A and B.
4. C will exhibit monophony if A and B are monophonic.

4.1 Overview of melody morphing method

The meaning of morphing is to change one image into another through a seamless transition. For example, a morphing method for a facial image can create intermediate images through the following operations.

1. Link characteristic points such as the eyes and nose, in two images as shown in Fig. 10a.
2. Rate the intensities of shape (position), color, and so forth in each image.

3. Combine the images.

Similarly, our melody morphing method creates intermediate melodies through the following operations.

1. Link the most common information of the time-span trees of two melodies, as shown in Fig. 10b.
2. Abstract the notes of a melody in a differing branch of the time-span tree by using the melody divisional reduction step of our melody morphing method.
3. Combine both melodies.

Figure 11 illustrates the melody morphing method.

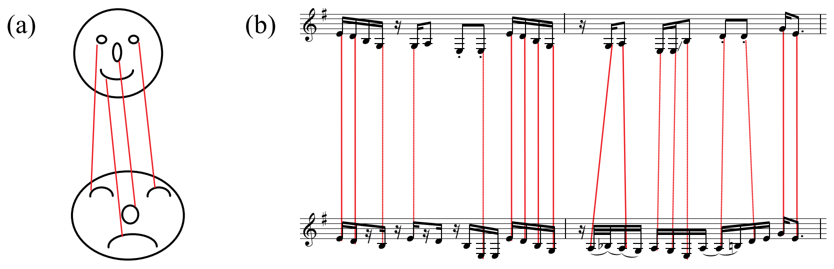

Fig. 10 Examples of linking two images and two melodies

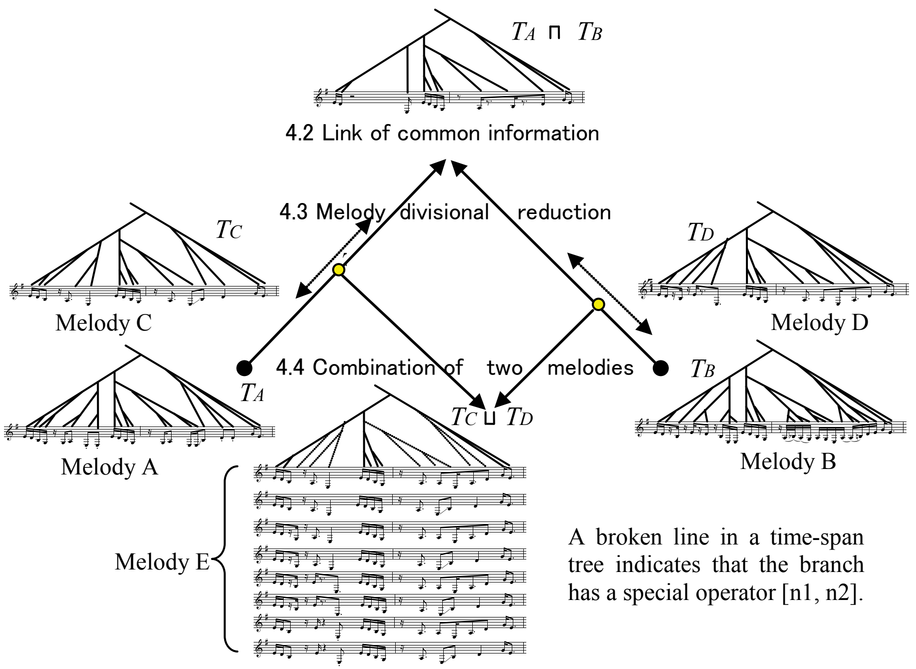
$T_A \sqcap T_B$
4.2 Link of common information
4.3 Melody divisional reduction
T_C
Melody C
T_D
Melody D
4.4 Combination of two melodies
T_A
Melody A
$T_C \sqcup T_D$
T_B
Melody B
Melody E
A broken line in a time-span tree indicates that the branch has a special operator [n1, n2].

Fig. 11 Overview of the melody morphing method

4.2 Linking of common information

By using the respective time-span trees T_A and T_B from melodies A and B, we can calculate the most common information $T_A \sqcap T_B$, which includes not only the essential parts of A but also those of B. The meet operation $T_A \sqcap T_B$ abstracts common notes from T_A and T_B, and the discarded notes are then regarded as the difference information of T_A and T_B.

When calculating $T_A \sqcap T_B$ by extracting the largest common part of T_A and T_B in a top-down manner, the result may change depending on whether octave notes such as C4 and C3 can be distinguished. If we discriminate octave notes, then C4 $\sqcap$C3 will be empty, denoted as $\perp$. On the other hand, if we do not discriminate octave notes, the result is just C, which abstracts the octave information. Here, we regard a note and its octave as different notes, because processing is difficult if the octave information is not defined.

4.3 Melody divisional reduction

We next consider that the difference information of T_A and T_B includes features not present in the other respective melody. Therefore, we need a method for smoothly inserting or removing such features. The melody divisional reduction step of our melody morphing method abstracts the notes of the melody in a difference branch of the time-span tree by applying the abstraction described in Section 3.1.

With this method, we can acquire melodies *Cm* (m=1,2,…,n) from T_A and $T_A \sqcap T_B$ with the following algorithm. The subscript m indicates the number of notes in the difference information of the time-span trees that are included in T_{Cm} and not included in $T_A \sqcap T_B$.

Step 1: Determine the level of abstraction.
The user selects a parameter L that determines the level of abstraction of the melody. The L can range from 1 to the number of notes in the difference information of the time-span trees that are included in T_A but not included in $T_A \sqcap T_B$.

Step 2: Abstract notes in the difference information.
The note with the fewest dots in the difference information is selected and abstracted. The numbers of dots can be acquired from the GTTM analysis results [4]. If two or more notes share the fewest dots, we select the first one reading the music left to right.

Step 3: Iterate.
Step 2 is iterated L times.

Subsumption relations hold as follows for the time-span trees T_{Cm} constructed with the above algorithm:

$$T_A \sqcap T_B \sqsubsetneq T_{Cn} \sqsubsetneq T_{Cn-1} \sqsubsetneq \ldots \sqsubsetneq T_{C2} \sqsubsetneq T_{C1} \sqsubsetneq T_A . \qquad (2)$$

In Fig. 11, nine notes are included in T_A but not in $T_A \sqcap T_B$. Therefore, the value of n is 8, and we can obtain eight intermediate melodies *Cm* (m=1,2,…,n) between T_A and $T_A \sqcap T_B$. Hence, melody *Cm* attenuates features that occur only in melody A but not in B. Figure 12 illustrates this process.

In the same way, we can obtain melody D from T_B and $T_A \sqcap T_B$ in the following manner:

$$T_A \sqcap T_B \sqsubsetneq T_D \sqsubsetneq T_B . \qquad (3)$$

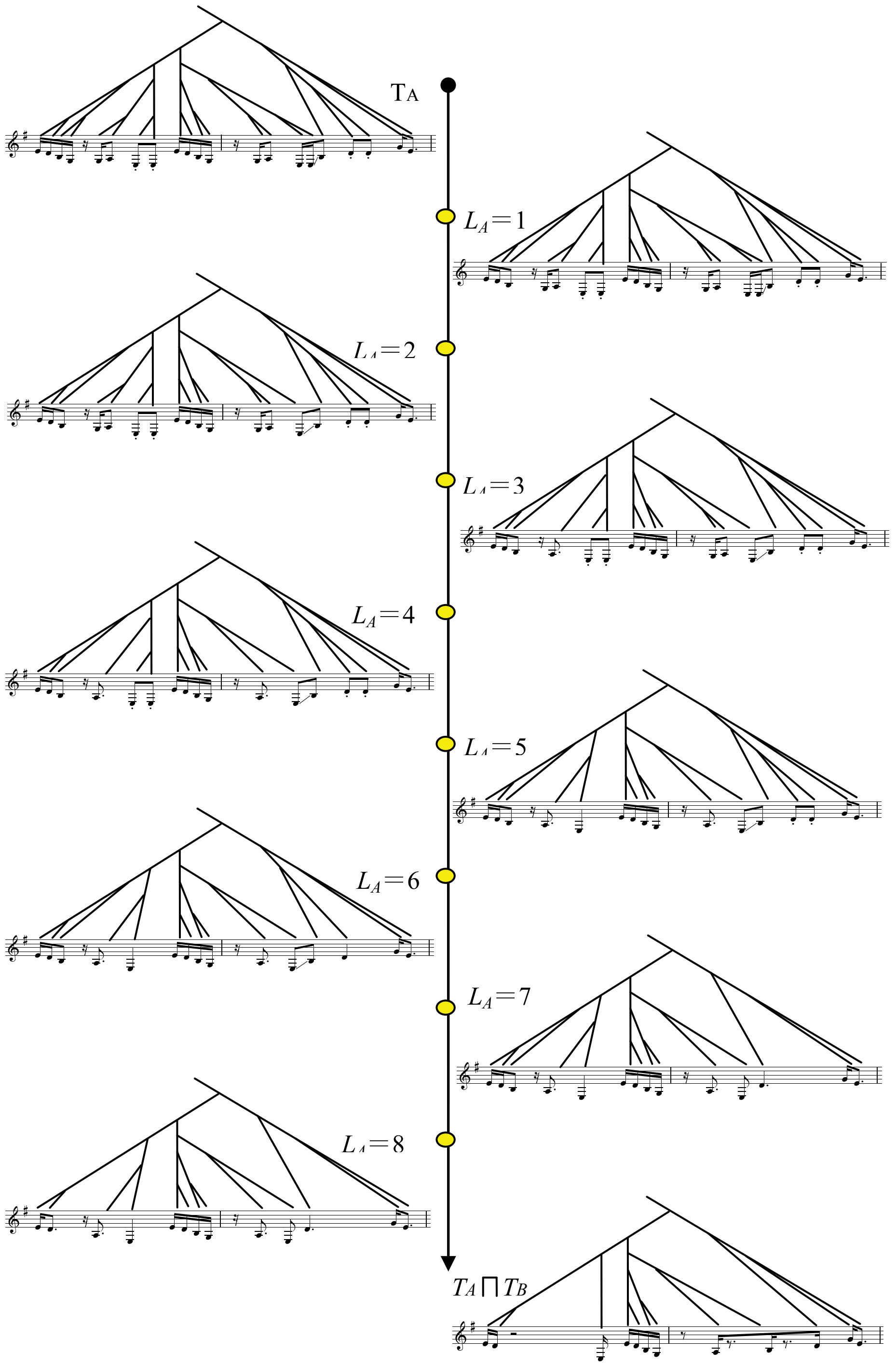

Fig. 12 Melody divisional reduction

4.4 Combination of two melodies

Finally, we use the join operator to combine melodies C and D, which are the results of divisional reduction using the time-span trees of melodies A and B. The simple join operator is not sufficient for combining T_C and T_D because $T_C \sqcup T_D$ does not always exhibit monophony even if T_C and T_D are monophonic. In other words, the result of the operation has chords when the time-span structures override and the pitches of the notes are different; therefore, the result violates condition 4 listed at the beginning of Section 4.

To solve this problem, we introduce a special operator [n1, n2], which indicates either note n1 or note n2, as a result of n1 $\sqcup$n2. Then, the result of $T_C \sqcup T_D$ is all monophonic combinations given by the operators.

5 Manual Generation of Intermediate Melody

Because of the problems described below with generating intermediate melodies on a computer with the method discussed in Section 4, we also had a musicologist manually generate a morphing melody.

5.1 Time-span analysis of two input melodies

Before linking the common information of two input melodies, we need their time-span trees. Although we have already developed grouping and metrical structure analyzers based on deep learning [3, 4], we have not yet developed a high-performance time-span tree analyzer [2]. Acquisition of time-span trees thus requires manual analysis by a musicologist.

If two input melodies are completely different, then their common information will be empty, $\perp$, making it difficult to generate intermediate melodies. Therefore, the two input melodies must be carefully chosen by the musicologist. The musicologist selected a variation and theme from Mozart's "Twelve Variations" (K. 265/300e) for the melodies, as explained in Section 6.

5.2 Time-span reduction

At present, the melody divisional reduction step of the melody morphing method cannot be automated. This is because the height of each branch in the time-span tree is not given, and the order of the notes to be reduced cannot be determined. A musicologist who can analyze time-span trees, however, can easily generate reduced melodies by considering the inverse process of composition depending on his/her interpretation, as described in Section 3.2. When multiple composition processes are possible, we consider all interpretations then select one.

5.3 Selection of note from special operators

By selecting a note from each special operator after combining two melodies, the output melody will be monophonic when the two input melodies are monophonic. Therefore, depending on how this note is selected, the melody changes. When the intermediate melody is unnatural, the musicologist adds notes to correct the melody. The details of these added notes are described in the following section on the experimental results.

6 Experimental Results

We asked a musicologist to generate intermediate melodies. The musicologist had over 10 years of experience in GTTM analysis and a deep understanding of the melody-morphing method.

For the input melodies, the musicologist selected the theme and Variation No. 1 of Mozart's Twelve Variations on "Ah vous dirai-je, Maman", K. 265/300e. She constructed nine intermediate melodies between the two melodies. Figures 13 and 14 show the first and second sets of four bars, respectively, for the theme, Variation No. 1, and 9 intermediate melodies, out of 40 bars total. In the figure, nonparenthetical notes were generated with the melody morphing method, while parenthetical notes were added by the musicologist. For each melody, Table 1 lists the total number of notes, number of notes generated with the melody morphing method, and number of notes added by the musicologist.

There were the following five processes for adding notes.

a) Adding appoggiaturas, auxiliary notes, and passing notes.
b) Borrowing a note from a neighboring branch or swapping the order of notes in a branch.
c) Dividing a note into two notes of the same pitch.
d) Expanding or contracting the melody.
e) Quoting a melody.

In Fig. 13, morphing melodies 3 and 5 are examples of adding passing notes, denoted as (a), while (b) is an example of swapping the order of notes from Variation No. 1. In Fig. 14, (c) is an example of dividing a note into two notes of the same pitch. Back in Fig. 13, (d) is an example of expanding a melody, specifically the last two notes in the fourth bar of Variation No. 1. Finally, in Fig. 14, (e) is an example of quoting a melody from the fifth and sixth bars of Variation No. 1.

All the notes in morphing melodies 1 and 2 were generated with the method because both melodies were close to the common information of the theme and Variation No. 1, so melodies could be generated by simply choosing an appropriate note from each special operator without adding any sound. All the notes in morphing melody 5 were also generated with the method because it was very close to a melody obtained by divisional reduction of Variation No. 1, so all the notes were eighth notes.

Among morphing melodies 1, 2, and 3, the three melodies closest to the theme, only 1.5% of the notes were added by the musicologist. In contrast, morphing melodies 8 and 9, the closest to Variation No. 1, had 37.9% of notes added by the musicologist. It was necessary to add notes to prevent unnatural melodies when generating multiple melodies between a melody with many eighth notes, such as morphing melody 5, and a melody with many sixteenth notes, such as Variation No. 1.

In total, 78.5% of the notes were generated with the melody morphing method, while the remaining 21.5% was added by the musicologist.

Table 1. Numbers of notes in each melody

	Total number of notes	Number of notes generated with melody morphing method	Number of notes added by musicologist
Theme	82	-	-
Morphing melody 1	51	51 (100%)	0 (0%)
Morphing melody 2	51	51 (100%)	0 (0%)
Morphing melody 3	93	90 (96.8%)	3 (3.2%)
Morphing melody 4	121	96 (79.3%)	25 (20.7%)
Morphing melody 5	94	94 (100%)	0 (0%)
Morphing melody 6	157	129 (82.2%)	28 (17.8%)
Morphing melody 7	176	157 (89.2)	19 (10.8%)
Morphing melody 8	253	164 (64.8%)	89 (35.1%)
Morphing melody 9	267	159 (59.6%)	108 (40.4%)
Variation No. 1	271	-	-

7 Conclusion

We used our melody morphing method based on the GTTM for composition. The following three points are the main contributions of this study.

- Enable manual generation of intermediate melodies
  We made melody morphing possible by combining our melody-morphing method based on the GTTM with a manual process.
- Make melody morphing available for composition
  Experimental results were obtained by having a musicologist compose with the melody morphing method. In the composed pieces, 78.5% of the notes were generated with the melody morphing method, while the remaining 21.5% was added by the musicologist.
- Consider various arranging processes
  We confirmed that the arranging processes used in the experiment could be classified into five types.

We plan to develop a support system for adjustment using the melody morphing method. We also plan to analyze and morph pieces other than classical pieces, such as jazz.

Acknowledgements

This work was supported by JSPS KAKENHI, Grant Numbers 17H01847 and 16H01744.

References

1. Lerdahl, F., and Jackendoff, R.: *A generative theory of tonal music*. Cambridge, Massachusetts: MIT Press (1983).
2. Hamanaka, M., Hirata, K., and Tojo, S.: Implementing a Generating Theory of Tonal Music, *Journal of New Music Research* (JNMR), vol. 35, no. 4, pp. 249-277 (2007).
3. Hamanaka, M., Hirata, K., and Tojo, S.: deepGTTM-I: Local Boundaries Analyzer Based on Deep Learning Technique, *13th International Symposium on Computer Music Multidisciplinary Research* (CMMR2016), pp. 8-20 (July 2016).
4. Hamanaka, M., Hirata, K., and Tojo, S.: deepGTTM-II: Automatic Generation of Metrical Structure Based on Deep Learning Technique, *13th Sound and Music Conference* (SMC2016), pp. 221-249 (2016).
5. Hamanaka, M., Hirata, K., and Tojo, S.: Melody Morphing Method Based on GTTM. In: *Proceedings of the 2008 International Computer Music Conference* (ICMC2008), pp. 155-158 (2008).
6. Hamanaka, M., Hirata, K., and Tojo, S.: Melody Extrapolation in GTTM Approach. In: *Proceedings of the 2009 International Computer Music Conference* (ICMC2009), pp. 89-92 (2009).
7. Papadopoulos, A., Roy, P., and Pachet, F.: Avoiding Plagiarism in Markov Sequence Generation. In: *Proceedings of AAAI-14*, pp. 2731-2737 (2014).
8. Herremans, D., and Sörensen, K.: Composing first species counterpoint with a variable neighbourhood search algorithm. *Journal of Mathematics and the Arts* 6(4), pp. 169-189 (2012).
9. Boulanger-Lewandowski, N., Bengio, Y., and Vincent, P.: Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription. In: *Proceedings of the 29th International Conference on Machine Learning* (ICML-12), pp. 1159-1166 (2012).
10. Roig, C., Tardón, L. J., Barbancho, I., and Barbancho, A. M.: Automatic melody composition based on a probabilistic model of music style and harmonic rules, *Knowledge-Based Systems* 71, pp. 419-434 (2014).
11. Bemman, B., and Meredith, D.: Generating Milton Babbitt's all-partition arrays. *Journal of New Music Research* 45(2), pp. 184-204 (2016).
12. Herremans, D., and Chew, E.: MorpheuS: Automatic music generation with recurrent pattern constraints and tension profiles, 2016 IEEE TENCON (2016).
13. Hirata, K., Tojo, S., Hamanaka, M.: Algebraic Mozart by Tree Synthesis, *Proceedings of Joint ICMC and SMC 2014*, pp. 991-997 (2014).
14. Hirata, K., and Aoyagi, T.: Computational Music Representation Based on the Generative Theory of Tonal Music and the Deductive Object-Oriented Database, *Computer Music Journal* 27(3), pp. 73-89 (2003).

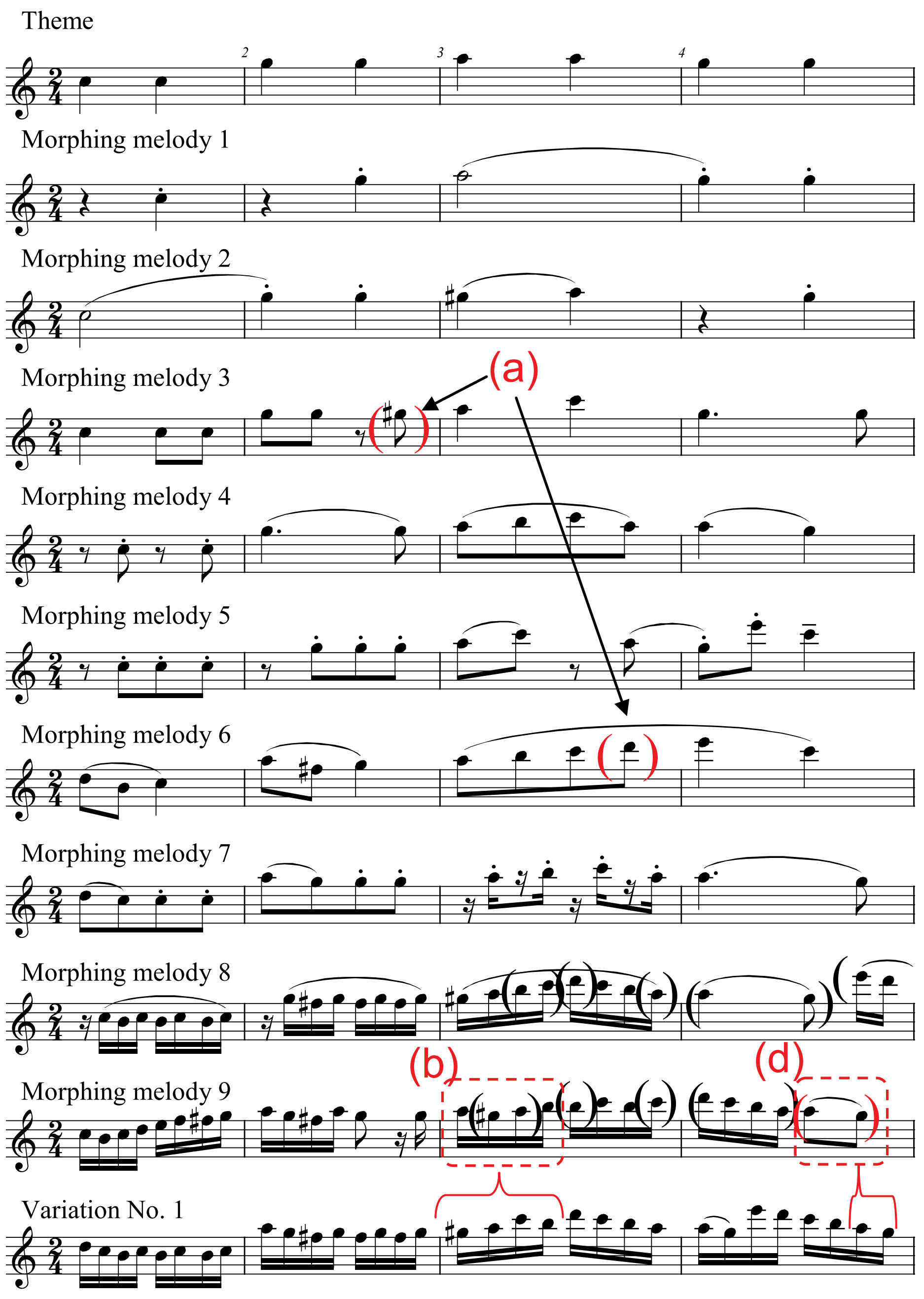

Fig. 13 First four bars of theme, Variation No. 1, and nine intermediate melodies

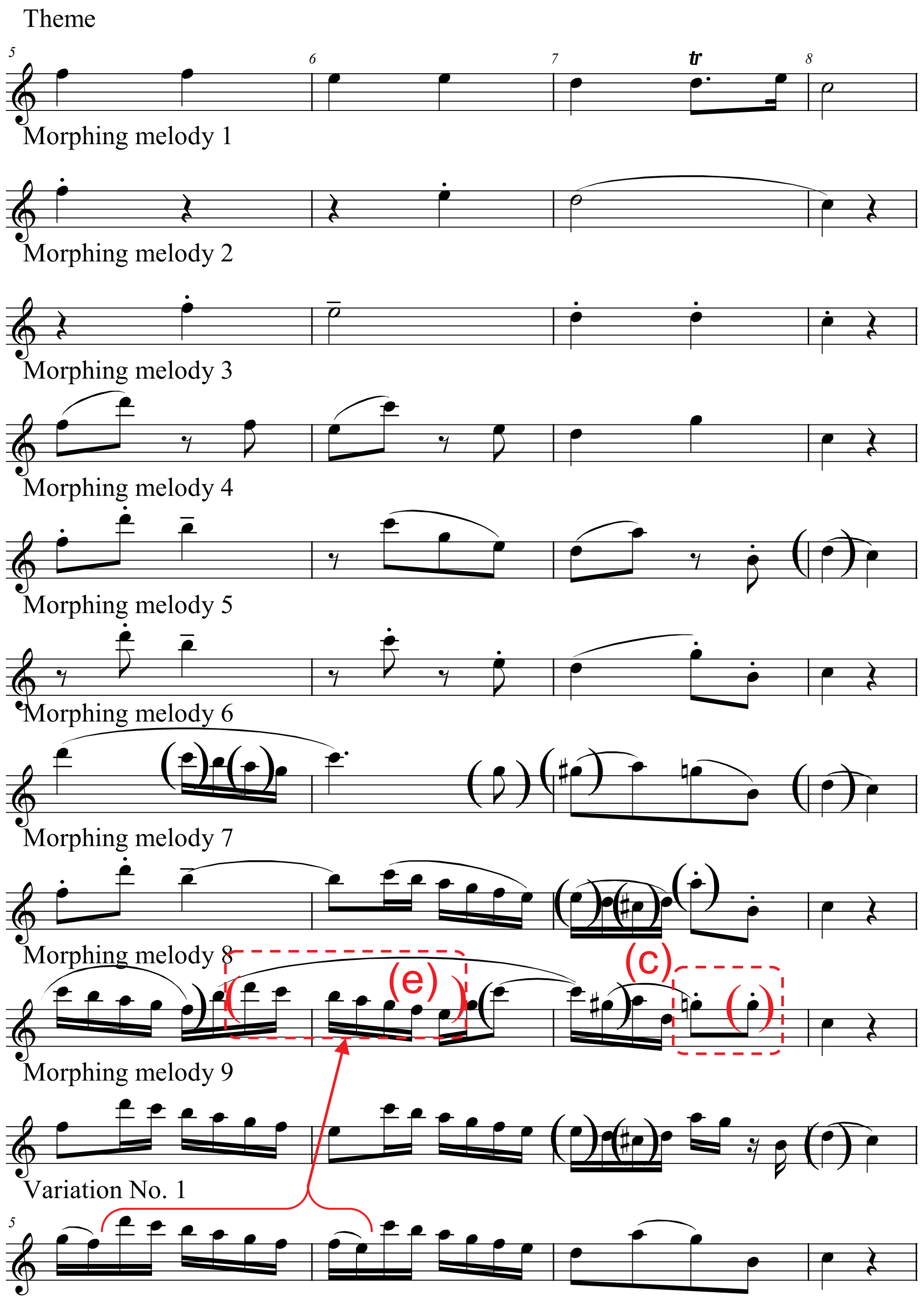

Fig. 14 Bars 5 to 8 of theme, Variation No. 1, and nine intermediate melodies.