METHOD TO DETECT GTTM LOCAL GROUPING BOUNDARIES BASED ON CLUSTERING AND STATISTICAL LEARNING

Kouhei Kanamori
University of Tsukuba
kouhei@music.iit.tsukuba.ac.jp

Masatoshi Hamanaka
Kyoto University
masatosh@kuhp.kyoto-u.ac.jp

Junichi Hoshino
University of Tsukuba
jhoshino@esys.tsukuba.ac.jp

ABSTRACT

In this paper, we describe σGTTM Ⅱ, a method that detects local grouping boundaries of the generative theory of tonal music (GTTM) based on clustering and statistical learning. It is difficult to implement GTTM on a computer because rules of GTTM often conflict with each other and cannot detect music structure as same manner. Previous methods have successfully implemented GTTM on a computer by introducing adjustable parameters or acquiring the priority of the rules by statistical learning. However, the values of the parameters and the priority of the rules are different depending on a piece of music. Considering these problems, we focused on the priority of the rules and we hypothesized that there are some tendency of rules which have more strong influence than other rules by the case of music. To ensure this hypothesis, we tried to classify each piece of music and tried to find the tendency of rules. Through the experiment, we found some tendency of rules and then we acquired some detectors which can analyze each piece of music more appropriately by reiterating clustering music and statistical learning.

1. INTRODUCTION

Our purpose of this research is to develop a music analysis system, which we call σGTTM Ⅱ, that can semi-automatically detect music structure based on the generative theory of tonal music (GTTM) by reiterating clustering and statistical learning [1]. In this paper, we describe how the local grouping boundaries of GTTM can be detected by choosing most appropriate detector.

GTTM is a music theory that enables comprehensive analysis of the structure of a piece of music, such as the grouping of melody (grouping structure) or the rhythm of music (metrical structure). GTTM analysis can also be used to obtain a time-span tree, which can express the priority of notes, thus enabling us to operate music structure deeply.

There has been previous research on using time-span tree to deeply analyze music structure [2], to realize musical expression [3−5] and to obtain abstracted melody [6]. However, due to GTTM's ambiguous rules, these studies [2−6] require a time-span tree that has already been made by musicologists.

In order to acquire this time-span tree in the viewpoint of computational music theory, there has been a study that proposed extended GTTM, called exGTTM, in which the ambiguity of GTTM rules is covered by parameterization. This exGTTM was implemented on a computer as an Automatic Time-span Tree Analyzer (ATTA) [7], which can acquire time-span tree by adjusting parameters. ATTA enables us to interpret music structure more flexibly by adjusting parameters, but adjusting the parameters is difficult because there are so many of them.

In another study, 100 pieces of music structure data were analyzed by a musicologist on the basis of GTTM and to identify the priority of the rules of GTTM by statistical learning. This system is called σGTTM, which can detect local grouping boundaries automatically [8]. However, this system sometimes outputs unnatural local grouping boundaries because there are a lot of tendencies of being local grouping boundaries and this system could reflect only one tendency among them.

To overcome these problems, the purpose of our research is to detect possible music structures automatically and then determine the most appropriate structure from among them by following method. First, we classify 100 pieces of music data into various clusters and then determine the priority of the rules per cluster by statistical learning. Next, we again divided the data reiteratively into various clusters based on the priority of rules and constructed gradually the clusters and detectors of local grouping boundaries that best suited each piece of music. We think that the system should be able to choose potential music structures by user because we think that the user's preference of music structure should be reflected in the system. Experimental results demonstrated that the proposed system outperformed the previous system in choosing the most appropriate detector.

2. GTTM MUSIC THEORY AND ITS AMBIGUOUS RULES

The Generative Theory of Tonal Music (GTTM) was formed by F. Lerdahl and R. Jackendoff in 1983. GTTM

was constructed more strictly than other music theories and it can treat the structure of each piece of music comprehensively. However, when it comes to implementing it on a computer, there is problem with the ambiguity of its rules. In this section, we describe a method of analyzing music structure by GTTM in section 2.1 and discuss the problem with ambiguous rules in section 2.2.

2.1 Method of analyzing music structure in GTTM

In GTTM, there are four steps to analyze music structure:
Step 1: Analysis of grouping structure, in which music is divided into some groups.
Step 2: Analysis of metrical structure, in which the rhythm structure of music is detected.
Step 3: Analysis of time-span reduction, in which the priority of each note in the music is detected and then expressed in a tree structure.
Step 4: Analysis of prolongational reduction, in which the tension and relaxation structure of the music is expressed in a tree structure.
An example of analysis by GTTM is shown in Figure 1.

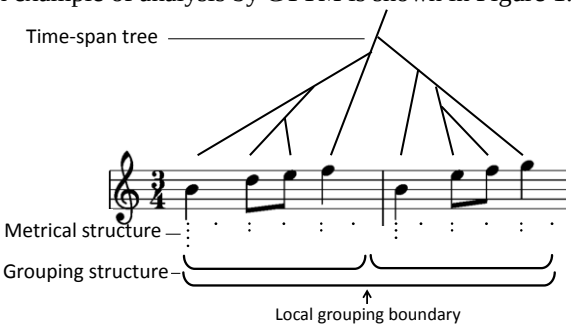

Figure 1. Example of analysis by GTTM.

Each step constructs well-formedness rules and preference rules. Well-formedness rules construct the initial framework of a music structure and preference rules detect more preferable music structures in the framework. Each music structure is hierarchical. The regular order of a GTTM music structure is constructed by analyzing each step. In this work, we treat the first step, grouping structure.

The well-formedness rules of the grouping structure are called grouping well-formedness rules (GWFR) and the preference rules are called grouping preference rules (GPR). GPR can classify two types of structure: one for treating the lowest (local) grouping structure (GPR1, 2, 3) and the other for treating the higher grouping structure. The interval between notes in which the local GPR is applied has the possibility of grouping boundary. An example of analyzing local grouping structure is shown in Figure 2.

2.2 Problems with ambiguous rules in GTTM

When we analyze music structure by GTTM, we may deal with a conflict between preference rules, since preference rules do not have any individual priority among themselves. Originally preference rules are formed to deal with human's preference, but that conflict between preference rules causes some difficulty when it comes to implementing GTTM on a computer. At that situation, we think that the analysis of the local grouping structure we treat in this paper has mainly two problems.

The first problem is conflict between rules. In the example of analysis shown in Figure 2, GPR2a and GPR2b are applied between notes 17 and 18 and GPR3a is applied between notes 18 and 19. GPR2a is applied when there is a rest or at the end of a slur, GPR2b is applied when there is a relatively higher duration, and GPR3a is applied when there is relatively higher difference of pitch between notes. In this case, we cannot detect that both 17−18 and 18−19 are grouping boundaries because of GPR1, which means that the grouping of one note must be avoided. This means we have to choose either 17−18 or 18−19 as the boundary, but in GTTM there are no rules for making this choice.

The second problem is that grouping boundary is not always applied in the same manner as local GPR. In Figure 2, GPR3a is applied between 5−6 and 23−24 and there are local grouping boundaries, but there is no local boundary in spite of the presence of GPR3a in 11−12 and 18−19 and 29−30 and 32−33. This problem cannot be resolved in GTTM.

3. CONSTRUCTION OF σ GTTM Ⅱ

We hypothesize that each piece of music has some tendency about priority of GTTM rules, which mean local grouping preference rules (local GPR) in this paper. If we can find that priority from some analysis of GTTM, we can analyze each piece of music more appropriately. Thus we use statistical learning for extracting that priority. Considering that the priority of local GPR cannot find until applying statistical learning, we reiteratively classify each piece of music into various clusters and construct detectors of local grouping structure gradually by applying statistical learning per cluster.

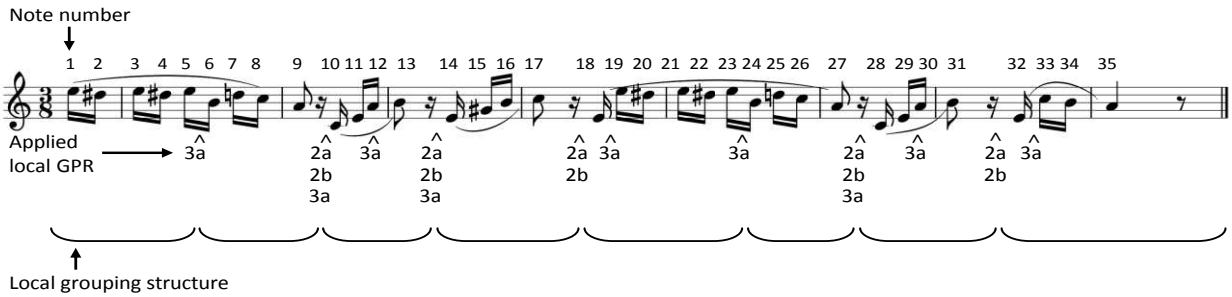

Figure 2. Example of analyzing local grouping structure.

In this section, we give an overview of proposed system σ GTTM Ⅱ in section 3.1, describe the method of detecting local grouping structure in section 3.2 and the method of detecting priority of local GPR used in previous research σ GTTM in section 3.3.

3.1 Overview of σ GTTM Ⅱ

Figure 3 shows an overview of proposed system σ GTTM Ⅱ. The main idea of this system is to reiterate clustering and statistical learning in order to classify each piece of music on the basis of priority of local GPR and detect local grouping structure more appropriately and easily. This system can classify each piece of music into some clusters and output detector of local grouping structure per cluster. This means the system outputs some candidates about local grouping structure reflected various priority of local GPR. Users can detect local grouping boundary more easily by choosing most preferable detector from among some candidates.

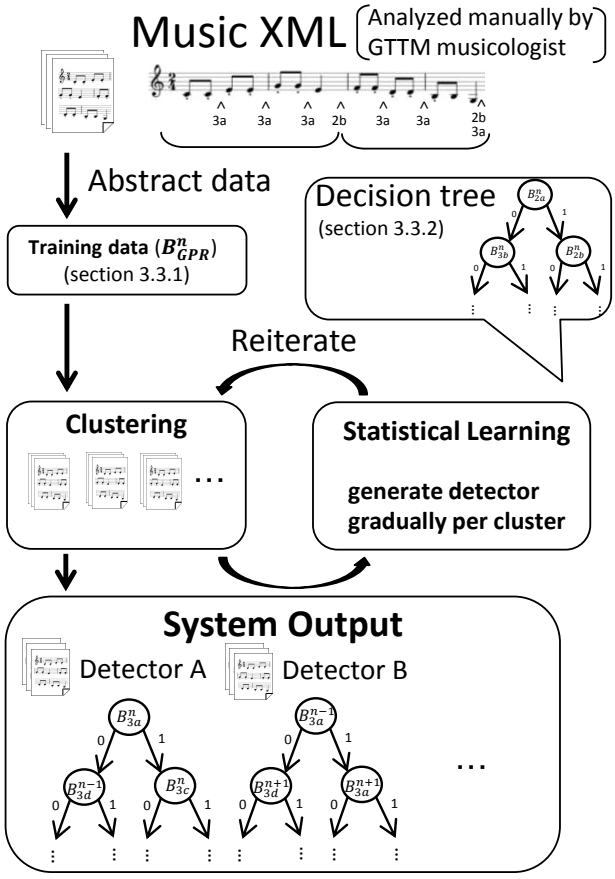

Figure 3. Overview of σ GTTM Ⅱ.

3.2 Method of detecting local grouping structure

The way to detect local grouping structure is to choose the detector you most preferred. When this system is used to musicXML data which is not trained, system outputs some candidates of local grouping structure about that data by some detectors which were already constructed. Users can see there detectors, which mean you can see the priority of local GPR of each candidates about local grouping structure. The main reason of designing this system as flexible detecting method of local grouping structure is to reflect user's preference about local grouping structure. Preference is different by each user, so this system outputs the some detectors, which are designed to reflect various tendencies about priority of local GPR.

3.3 Method of detecting priority of local GPR used in previous research σ GTTM

In this work, we use method of obtaining abstracted data (training data) and detecting priority of local GPR used in previous research σ GTTM. In this subsection, we give an overview of the method of abstracting musicXML in subsection 3.3.1 and decision tree in subsection 3.3.2 and detecting priority of local GPR in subsection 3.3.3.

3.3.1 Training data

100 musicXML was chosen as training data data of each piece of music, which is analyzed by GTTM musicologist manually and checked by GTTM experts. The objective value we want to know is the existence of local grouping boundary (is shown as b) so the value can be represented by 1 or 0 (boundary exists or not). Local GPR also should be abstracted because whether there is a boundary or not is decided by the local GPR. Considering that there is a rule in local GPR of avoiding single note grouping, not only the checking interval n (between note n and note n+1), but also neighbor interval (interval n-1 and interval n+1) should be checked. So the data was abstracted by the form of B^n_{GPR}. The superscript n means the checking interval n. The subscript GPR means the kind of local GPR. Local GPR we treat are 6 kinds (2a, 2b, 3a, 3b, 3c, 3d) so the abstracted data of checking interval can be shown as $B^n_{2a}, B^n_{2b}, B^n_{3a}, B^n_{3b}, B^n_{3c}, B^n_{3d}$. Considering the neighbor interval, the total abstracted data can be shown by 18 elements. These elements have a value 1 or 0 (rules exist or not). By the 18 elements the existence of local grouping boundary (b) is decided.

3.3.2 Decision Tree

Decision tree is one of statistical learning method. It can represent objective value and the priority of making decision as easy way to understand. It consists of mainly leaves and branches and ramification and this tree is upside down. The principle to make decision is due to the value of each ramification. Through this decision tree learning, the more the kind of ramification has influence to making decision, the more that ramification get near to root position.

3.3.3 Detecting priority of local GPR by decision tree

We chose C4.5, an algorithm developed by J. R. Quinlan [9] to construct the decision tree. Figure 4 shows an example of the constructed decision tree. From training data, we can obtain the conditional probability of local grouping boundaries for each combination of local GPR. When

this conditional probability is 0.5 or more, we detect to exist a local grouping boundary (b = 1), and when it is less than 0.5, we do not detect to exist boundary (b = 0). For the example in Figure 4, we detect a local grouping boundary exists in the case of $B_{2a}^{n} = 1\ and\ B_{2b}^{n} = 1$.

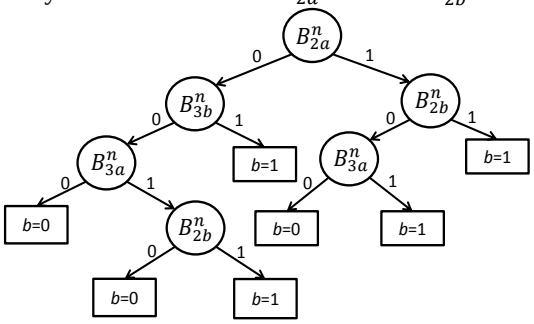

Figure 4. Example of constructed decision tree.

4. METHOD OF CLUSTERING MUSIC

To classify each piece of music on the basis of priority of local GPR, we reiterated clustering and statistical learning and generated detectors gradually. Figure 5 shows the details of clustering method. In this section, we describe the details of this method in section 4.1, method of evaluating each piece of music in section 4.2 and discuss about number of clusters in section 4.3.

4.1 Details of clustering method

First, we classify training data into some clusters. The training data of each cluster is then trained by a decision tree. After this training, a decision tree of GPR priority is constructed. Detector means the constructed decision tree. In figure 5 cluster and detector A, B, etc. means detector A is constructed in cluster A and detector B is constructed in cluster B, etc. However, this part is problematic because an irrelevant analyzed music structure might exist in the same cluster due to the detectors of each cluster representing tendency of the entire music structure as the same for each cluster.

To solve this problem, the system individually evaluates the performance of each detector as they are constructed and then reclassifies the training data into clusters which generated most performed detector. In figure 5 the clusters after reclassified are represented as A', B', etc. And then system compares the training data of each cluster between before (A, B, etc.) and after reclassification (A', B', etc.). The less the training data in the cluster changes, the more the detectors that are constructed cover the tendency of the priority of local GPR of all training data in the cluster.

After this comparison between clusters, if the total difference of training data before and after reclassification is more than two, the system returns to constructing detectors again and if the total difference is less than one, or if reclassification has been performed 150 times, the system outputs training data and detectors of each cluster. Finally, we construct the most appropriate detector on the basis of the priority of local GPR of the entire training data in a cluster.

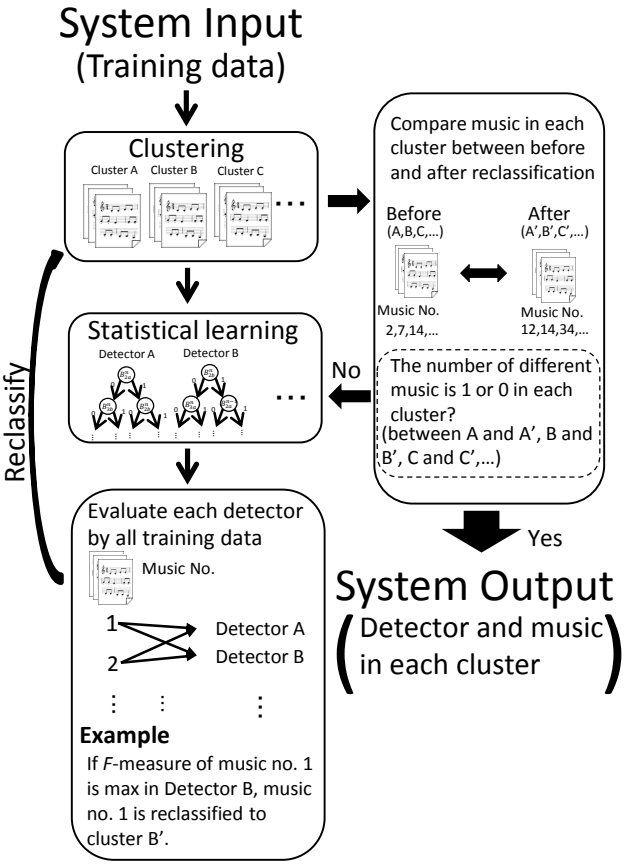

Figure 5. details of clustering method.

4.2 Method of evaluating each detector

When the system reclassifies training data, system evaluates each detector by F-measure, which consists of precision (P) and recall (R). Precision is the ratio of corresponding to correct local grouping boundary in the output of the system. Recall is the ratio of corresponding to the output of the system in the correct local grouping boundary. The F-measure is represented as

$$F_{measure} = 2 \times \frac{P \times R}{P + R} \quad (1)$$

Each training data is reclassified by cluster which generates most high performed detector.

4.3 Number of clusters

When we classify each piece of music into some clusters at first, we don't know how many tendencies each piece of music has. So, we changed the number of cluster at first classification from 1 to 100. This means at first the number of input cluster of this system is 1 and system outputs 1 detector, and then number of input cluster is 2 and system outputs 2 detectors. Thus the system runs 100 times through the input and output. At each runtime the system reiterating clustering and statistical learning many times until it gets ready to output detectors.

5. EXPERIMENTAL RESULTS

We implemented proposed system σGTTMⅡ and evaluated the performance of detectors constructed in clusters by detecting the local grouping boundaries of 100 piece of music. Figure 6 shows the results of this experiment. The precision is the value when most appropriate detector was chosen.

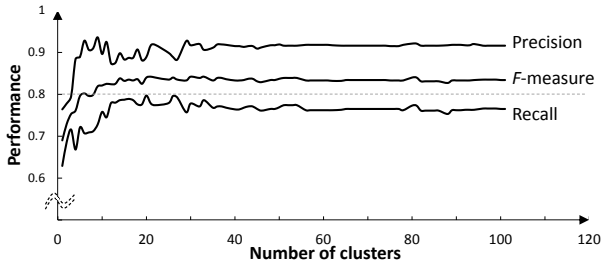

Figure 6. Performance of σGTTMⅡ.

As the initial clusters grew bigger, some clusters in which the number of music became 0 appeared, so there were some cases in which the number of clusters outputted by the system differed from the initial clusters. The relationship between the number of initial clusters and the output clusters is shown in Figure 7.

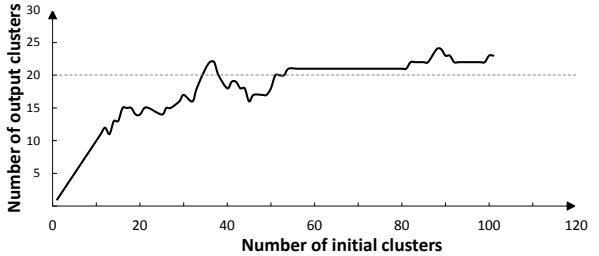

Figure 7. Transition of cluster numbers.

Next, we compared σGTTMⅡ with previous research ATTA and σGTTM. The performance of σGTTMⅡ is the value when the number of clusters was 10. Results demonstrated that σGTTMⅡ outperformed the previous research when it comes to choosing most appropriate detector (Table 1). Precision and recall of ATTA cannot be mentioned because they were not described in the previous study [7]. Also F-measure of ATTA is evaluated under the situation of treating higher grouping structure.

	Precision *P*	Recall *R*	*F*-measure
ATTA (Parameters are adjusted)	—	—	0.77
σGTTM	0.764	0.630	0.690
σGTTMⅡ (Number of clusters is 10)	0.905	0.734	0.811

Table 1. Evaluation experiment (closed).

To determine the performance of σGTTMⅡ with data that is not trained, we evaluated this system using correct data that was analyzed by a GTTM musicologist and checked by three GTTM experts. We also evaluated previous research σGTTM in the same situation for comparison. Results show that σGTTMⅡ outperformed previous research σGTTM when it comes to choosing most appropriate detector in the case of no trained data (Table 2).

	Precision *P*	Recall *R*	*F*-measure
σGTTM	0.467	0.736	0.571
σGTTMⅡ (Number of clusters is 10)	0.684	0.916	0.783

Table 2. Evaluation experiment (open).

6. CONCLUSION

In this paper, we described a method of semi-automatically detecting the local grouping boundaries of GTTM by choosing the appropriate detector. In this method, we avoid conflicting GPR rules by using a decision tree and detecting local GPR priorities. Moreover, we divide training data that have similar priorities of local GPR into various clusters and construct the detectors most appropriate for each cluster. Experimental results demonstrated that the proposed system outperforms a previous system when it comes to choosing the most appropriate detector.

We expected that each piece of music in same cluster has same feature about part of musical piece, but we couldn't find them at that point. Our next step is to try to find some same feature in each piece of music in same cluster. Also we try to extend this system to higher grouping structure, metrical structure, and time-span reduction.

7. REFERENCES

[1] F. Lerdahl and R. Jackendoff, "A Generative Theory of Tonal Music", Cambridge: The MIT Press, 1983.

[2] K. Hirata and T. Aoyagi, "Computational Music Representation based on the Generative theory of Tonal Music and the Deductive Object-Oriented Database", Computer Music Journal 27(3), 73−89, 2003.

[3] N. Todd, "A Model of Expressive Timing in Tonal Music", Musical Perception, 3:1, 33−58, 1985.

[4] G. Widmer, "Understanding and Learning Musical Expression", Proceedings of International Computer Music Conference, pp. 268−275, 1993.

[5] K. Hirata and R. Hiraga, "Ha-Hi-Hun plays Chopin's Etude", Working Notes of IJCAI-03 Workshop on Methods for Automatic Music Performance and their Applications in a Public Rendering Contest, pp. 72−73, 2003.

[6] K. Hirata and S. Matsuda, "Interactive Music Summarization based on GTTM", In Proc. of ISMIR 2002, pp. 86−93, 2002.

[7] M. Hamanaka, K. Hirata and S. Tojo, "ATTA: Automatic Time-span Tree Analyzer based on Extended GTTM", Proceedings of the 6th International Conference on Music Information Retrieval conference (ISMIR2005), pp. 358−365, September 2005.

[8] Y. Miura, M. Hamanaka, K. Hirata, and S. Tojo, "Use of Decision Tree to Detect GTTM Group Boundaries", Proceedings of the 2009 International Computer Music Conference (ICMC2009), pp. 125−128, August 2009.

[9] J. R. Quinlan, "C4.5: Programs for Machine Learning", Morgan Kaufman Publishers, 1993.