TREE-STRUCTURED PROBABILISTIC MODEL OF MONOPHONIC WRITTEN MUSIC BASED ON THE GENERATIVE THEORY OF TONAL MUSIC

Eita Nakamura[1], *Masatoshi Hamanaka*[2], *Keiji Hirata*[3], *Kazuyoshi Yoshii*[1]

[1]Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
[2]Graduate School of Medicine, Kyoto University, Kyoto 606-8507, Japan
[3]Future University Hakodate, Hakodate 041-8655, Japan

ABSTRACT

This paper presents a probabilistic formulation of music language modelling based on the generative theory of tonal music (GTTM) named probabilistic GTTM (PGTTM). GTTM is a well-known music theory that describes the tree structure of written music in analogy with the phrase structure grammar of natural language. To develop a computational music language model incorporating GTTM and a machine-learning framework for data-driven music grammar induction, we construct a generative model of monophonic music based on probabilistic context-free grammar, in which the time-span tree proposed in GTTM corresponds to the parse tree. Applying the techniques of natural language processing, we also derive supervised and unsupervised learning algorithms based on the maximal-likelihood estimation, and a Bayesian inference algorithm based on the Gibbs sampling. Despite the conceptual simplicity of the model, we found that the model automatically acquires music grammar from data and reproduces time-span trees of written music as accurately as an analyser that required elaborate manual parameter tuning.

Index Terms— Statistical music language model, GTTM, PCFG, time-span tree analysis, statistical grammar induction

1. INTRODUCTION

Music transcription is a challenging topic in music processing. For this, acoustic modeling of musical sounds has been widely studied [1]. However, this is not sufficient to remove all ambiguities of transcription due to acoustic variations, and using prior knowledge on the output musical score is in order. The same situation happens in speech recognition, where the knowledge on linguistics is incorporated in the language model. The purpose of this study is to construct a model that can capture the grammar of written music.

Musical pieces have a hierarchical structure. Notes are grouped into motives, which are grouped into phrases, which are grouped into musical sentences (or passages), which are grouped into sections, etc. In addition, some notes are more salient than adjacent notes. This structure of music resembles the phrase structure of natural language [2], which can be represented as a syntax tree. The generative theory of tonal music (GTTM) [3] attempts to model such a hierarchical structure of music. GTTM proposes to use a tree representation called the time-span tree to describe the relative importance of notes. The time-span tree can be interpreted as a simplification or reduction process of a musical passage (Fig. 1). GTTM also proposes rules to derive a reasonable time-span tree for a given musical passage, which combine musical knowledge such as counterpoint theory [4], harmonic theory [5], and Schenkerian analysis [6].

This work is partially supported by JSPS KAKENHI Nos. 24220006, 26700020, 26280089, and 15K16054, and JST CREST.

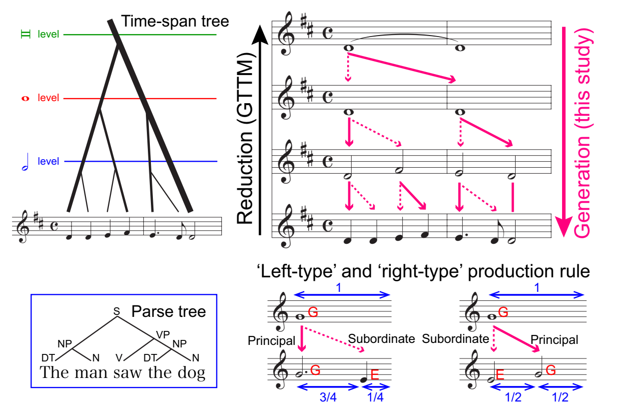

Fig. 1. Example of time-span tree and the reduction and generation of a musical passage (top). The two types of production rules in the probabilistic GTTM (bottom).

The computational formulation of GTTM has been studied in the last decades [7–10] because the prescribed rules of GTTM are not quantitatively or computationally formalised. Based on parameterisation of the rules in GTTM, an automatic time-span tree analyser (ATTA) was implemented [7]. The adjustment of the 46 parameters remained as a problem. Recently a probabilistic method for time-span tree analysis called σGTTM III is proposed, which can learn some parameters from labelled music data [10]. There are other notable studies on automatic music analysers [11, 12]. Although these studies provide grouping structure analysis, metrical analysis, and reduction analysis, they are not a generative model and cannot be directly applied to music transcription [1] or automatic music generation and arrangement [13]. Moreover, these studies have not taken advantage of the developed machine-learning techniques in natural language processing (NLP) [14–17].

To solve these problems, we formulate a probabilistic generative model of music based on GTTM named the probabilistic GTTM (PGTTM). We interpret that the time-span tree represents a production process of musical notes and formulate PGTTM with probabilistic context-free grammar (PCFG) [18, 19]. In contrast to the phrase structure grammar of linguistics where an intermediate node corresponds to grammatical categories such as 'NP' and 'VP', a node of the time-span tree represents a musical note (Fig. 1). In addition, two types of production rules must be distinguished to indicate the relative importance of children notes, and there are constraints on the pitches and note values (lengths) to consistently describe the reduction/generation process.

There are other works that applied PCFG models for music. The idea of using PCFG for the probabilistic formulation of GTTM is already explored in σGTTM III [10], which is not fully formulated as a generative model of musical notes and requires other analysing tools [7, 8] to obtain the time-span trees. A two-dimensional tree-structure model was proposed [20] and applied for multi-pitch analysis. Our study shares the same interests in unsupervised learning of music language model. A PCFG model for harmonic analysis was proposed in [21]. The main contribution of this study is to provide an explicit construction of a statistical music language model that integrates GTTM and a machine-learning framework for inducing music grammar from data. In this study, we confine ourselves to monophonic music.

We evaluate the proposed PGTTM in terms of the accuracy of time-span tree analysis. We examine the performance of the model with both supervised and unsupervised learning, and compare the results with previously proposed analysers. We found that the PGTTM performs as accurately as ATTA. This is encouraging as PGTTM is conceptually simple and does not require elaborate parameter tuning. Error analyses show that whereas local tree structures were relatively well estimated with the PGTTM, the higher-level structures were poorly estimated. The result suggests the importance of introducing latent grammatical categories into the PGTTM. Ideas for further refinements are given at the end.

2. PROBABILISTIC GTTM (PGTTM)

This section explains the proposed model.

2.1. Basic model

Each musical note is represented as a pair (p, r) of pitch p and note value r. We can use either spelled pitches or integral pitches, and the set of possible pitches is denoted by Ω_p. A rest is represented by a symbol 'R', which we also include in Ω_p. We define the note value as the score-written length of the note relative to a whole note (a quarter note has a $r = 1/4$, a dotted half note has a $r = 3/4$, etc.). The set of possible note values is denoted by Ω_r. Then a monophonic passage can be represented as a sequence $(p_n, r_n)_{n=1}^{N}$, where N is the number of notes and rests.

A PCFG model [18] is represented with the set of terminals Ω_T, the set of non-terminals Ω_N which contains the start symbol S, and the set of production rules R. In the Chomsky normal form, a production rule $A \to \alpha$ ($A \in \Omega_N$, $\alpha \in \Omega_N \times \Omega_N \cup \Omega_T$) is associated with a probability value $P(A \to \alpha)$. The symbol A and the symbols in α in a production rule will be called the parent and the children, respectively. Given a sequence of non-terminals $\boldsymbol{w} = w_1 \cdots w_N$ ($w_n \in \Omega_T$), a set of production rules to derive $\boldsymbol{w}$ from S is called a derivation of $\boldsymbol{w}$, which can be represented as a parse tree (Fig. 1).

In the probabilistic formulation of GTTM, we need the following modifications and extensions of this ordinary PCFG. First, the parent and the children of a production rule should be represented as musical notes. An exception is the most top production of notes from the start symbol, which will be explained in the next paragraph. Second, to describe the structure of the principal note vs. the subordinate note, we distinguish two types of production rules; one has the principal note on the left and the other has it on the right. Then a production rule can be written in the form $(p, r) \to s(p_L, r_L)(p_R, r_R)$ where $p, p_L, p_R \in \Omega_p$, $r, r_L, r_R \in \Omega_r$, and $s = L, R$ indicates that (p_s, r_s) is the principal note. Finally, since the time span must be conserved in producing a note, we must have $r = r_L + r_R$, and thus we can write $r_R = r - r_L$. The fact that the principal note has the same pitch as the parent note requires $p = p_s$ (Fig. 1).

Let us now define the PGTTM. The production process begins with the start symbol S with a note value r_S, which corresponds to the total note value of a musical passage. The production rules are either of the form $(S, r_S) \to s(p_L, r_L)(p_R, r_S{-}r_L)$ or $(p, r) \to s(p_L, r_L)(p_R, r{-}r_L)$ where $p, p_L, p_R \in \Omega_p$, $r, r_L \in \Omega_r$, and $s = L, R$. The associated probabilities are written as $P\big((S, r_S) \to s(p_L, r_L)(p_R, r_S{-}r_L)\big)$ and $P\big((p, r) \to s(p_L, r_L)(p_R, r{-}r_L)\big)$, which satisfy the following normalisation conditions:

$$\sum_{s,p_L,p_R,r_L} P\big((S, r_S) \to s(p_L, r_L)(p_R, r_S{-}r_L)\big) = 1, \quad (1)$$

$$\sum_{s,p_L,p_R,r_L} P\big((p, r) \to s(p_L, r_L)(p_R, r{-}r_L)\big) = 1. \quad (2)$$

There is a constraint that $P\big((p, r) \to s(p_L, r_L)(p_R, r{-}r_L)\big) = 0$ unless $p = p_s$ as explained above, and we also assume that rests must not be a principal note so that the probability is zero unless $p \in \Omega_p \backslash \{\mathrm{R}\}$. A set of production rules that yields a given musical passage is called a time-span tree.

The production probabilities strongly depend on the key of the musical passage, and this dependence can be theoretically described by first considering a different PGTTM for each of 24 keys (major keys and minor keys on 12 pitch classes) and then constructing their mixture model. In the following we consider a situation that the key is known in advance, so all passages can be transposed to C major or C minor.

2.2. Simplifications and refinements

Simplifications of the PGTTM described in Section 2.1 are necessary if we are to derive computationally tractable inference algorithms. First, we assume that the production rule probabilities are independent for pitch and note values. This means that the production rule probability is factorised as follows:

$$\begin{aligned} &P\big((p, r) \to s(p_L, r_L)(p_R, r{-}r_L)\big) \\ &\quad = P^s(s) P^p(p \to s p_L p_R) P^r(r \to s r_L(r{-}r_L)) \end{aligned} \quad (3)$$

where $\sum_s P^s(s) = 1$, $\sum_{p_L,p_R} P^p(p \to s p_L p_R) = 1$, and $\sum_{r_L} P^r(r \to s r_L(r{-}r_L)) = 1$, and similarly for the production rules from S. Second, we represent the pitch with an integral pitch class, and thus $\Omega_p = \{\mathrm{C}, \mathrm{C\#}, \cdots, \mathrm{B}, \mathrm{R}\}$ (where C#=D♭ etc.). We will use the following notations: $\phi_s = P^s(s)$, $\theta_{spp_Lp_R} = P^p(p \to s p_L p_R)$, $\rho_{srr_L} = P^r(r \to s r_L(r{-}r_L))$, and $\Theta = (\phi_s, \theta_{spp_Lp_R}, \rho_{srr_L})_{s,p,p_L,p_R,r,r_L}$. Finally, we assume that the probabilities for note values are scale invariant so that $P^r(r \to s r_L(r{-}r_L))$ is a function of r_L/r and not separately dependent of r and r_L. The probability of a time-span tree T is given as

$$P(T|\Theta) = \prod_{s,p,p_L,p_R,r,r_L} \big(\phi_s \theta_{spp_Lp_R} \rho_{srr_L}\big)^{c((p,r) \to s(p_L,p_R,r_L);T)}$$

where $c((p, r) \to s(p_L, p_R, r_L); T)$ is the number of times that the production rule $(p, r) \to s(p_L, r_L)(p_R, r{-}r_L)$ appears in T.

It has been noted that the choice of the principal note is influenced by the magnitude relation of metrical weights of the relevant note pair (Rule TSRPR 1 in [3]). Here, the metrical weight of a note indicates the relative strength of the metrical (beat) position of its onset [22]. To incorporate this preference rule, we allow that the probability P^s depends on the relative metrical weights of the relevant note pair. If we represent the metrical weights of note (p_L, r_L) and (p_R, r_R) as ω_L and ω_R, P^s take different values depending on whether ω_L/ω_R is less than, equal to, or greater than unity.

2.3. Bayesian formulation of the PGTTM

As we explain in Sec. 3, we can derive an unsupervised learning algorithm for PGTTM based on the maximal-likelihood principle. An alternative way of unsupervised learning is Bayesian inference [14]. With the Bayesian formulation, we can develop a Monte-Carlo method for inference that can learn parameters without getting stuck in local optima.

For Bayesian inference, we introduce a prior distribution for the probabilistic parameters of the PGTTM. Since the production-rule probabilities obey categorical distributions, Dirichlet distributions are their natural priors. Let $\boldsymbol{\eta} = (\eta_s)_s$, $\boldsymbol{\lambda}_{sp} = (\lambda_{spp_Lp_R})_{p_L,p_R}$, and $\boldsymbol{\nu}_{sr} = (\nu_{srr_L})_{r_L}$ denote the Dirichlet parameters for $\boldsymbol{\phi} = (\phi_s)_s$, $\boldsymbol{\theta}_{sp} = (\theta_{spp_Lp_R})_{p_L,p_R}$, and $\boldsymbol{\rho}_{sr} = (\rho_{srr_L})_{r_L}$, respectively. Then the prior distributions are given as $P_{\rm Dir}(\boldsymbol{\phi}|\boldsymbol{\eta})$ and similarly for $\boldsymbol{\theta}_{sp}$ and $\boldsymbol{\rho}_{sr}$, where $P_{\rm Dir}$ denotes the Dirichlet distribution.

3. INFERENCE ALGORITHMS FOR PGTTM

Basic inference algorithms for PCFG have been well-developed. With some modifications, we can derive inference algorithms for the PGTTM. This section summarises the results.

3.1. CYK-Viterbi, inside, and outside algorithms

Given a musical passage $W = (p_n, r_n)_{n=1}^N$, the most probable time-span tree can be obtained with the CYK-Viterbi algorithm, which is a dynamic programming. We will use the notation $W_n = (p_n, r_n)$, $W_n^m = W_n \cdots W_m$, and $r_n^m = r_n + \cdots + r_m$. We introduce a variable $\gamma_{nmp} = \max_T P(T)$ for $1 \le n \le m \le N$ and $p \in \Omega_p$ where T denotes a time-span tree with parent (p, r_n^m) and yield W_n^m (we express this as $(p, r_n^m) \overset{T}{\Rightarrow} W_n^m$). This can be calculated recursively as

$$\gamma_{n(n+\ell)p} = \max_{k,s,p^L,p^R} \Big\{ P\big((p, r_n^{n+\ell}) \to s(p^L, r_n^{n+k-1})(p^R, r_{n+k}^{n+\ell})\big) \cdot \gamma_{n(n+k-1)p^L} \gamma_{(n+k)(n+\ell)p^R} \Big\} \tag{4}$$

for all $p \in \Omega_p$ with the initial values $\gamma_{nnp} = \delta_{p,W_n}$ (δ denotes Kronecker's delta). Then the probability of the most probable time-span tree is given as γ_{1NS}. In the calculation of Eq. (4), the arguments that yield the maximal probability are also memorised as $\hat{k}_{n(n+\ell)p}$, $\hat{s}_{n(n+\ell)p}$, etc. Once all γ_{nmp} have been computed, we can obtain the most probable time-span tree by expanding the tree according to these arguments starting from (S, r_1^N), as in the back-tracking procedure of the Viterbi algorithm for hidden Markov model.

To learn parameters of the PGTTM, we calculate the inside variables $\beta_{nmp}(W)$ and outside variables $\alpha_{nmp}(W)$ defined as

$$\beta_{nmp}(W) = \sum_{T:(p,r_n^m) \overset{T}{\Rightarrow} W_n^m} P(T), \tag{5}$$

$$\alpha_{nmp}(W) = \sum_{T:(S,r_1^N) \overset{T}{\Rightarrow} W_1^{n-1}(p,r_n^m)W_{m+1}^N} P(T). \tag{6}$$

If the dependency on W is self-evident, we simplify the notation as β_{nmp} and α_{nmp}. These quantities can also be computed by dynamic programming algorithms called the inside and outside algorithms. For PGTTM, the inside algorithm has an update equation same as Eq. (4) except with the maximisation $\max_{k,s,p^L,p^R}$ replaced by a summation $\sum_{k,s,p^L,p^R}$. The outside algorithm is based on the following recursive equation:

$$\alpha_{nmp} = \sum_{s,q,p_R,k} P\big((q, r_n^k) \to s(p, r_n^m)(p_R, r_{m+1}^k)\big) \alpha_{nkq} \beta_{(m+1)kp_R} + \sum_{s,q,p_L,k} P\big((q, r_k^m) \to s(p_L, r_k^{n-1})(p, r_n^m)\big) \alpha_{kmq} \beta_{k(n-1)p_L} \tag{7}$$

with initial values $\alpha_{1NS} = 1$ and $\alpha_{1Np} = 0$ ($p \neq S$). In practice the left-hand side of Eq. (7) can be computed more efficiently by applying the constraints explained below Eq. (2).

3.2. EM algorithm for maximal-likelihood estimation

Given a set of (unlabelled) data W, the probability parameters can be estimated by the maximal-likelihood principle: $\hat{\Theta} = \operatorname{argmax}_\Theta P(W|\Theta)$. The expectation-maximisation (EM) algorithm can be derived by the following iterative minimisation [23]:

$$\begin{aligned} \Theta^{\rm new} &= \underset{\Theta}{\operatorname{argmax}}\, Q(\Theta, \Theta^{\rm old}) \\ &= \underset{\Theta}{\operatorname{argmax}} \sum_T P(T|W, \Theta^{\rm old}) \ln P(T, W|\Theta). \end{aligned} \tag{8}$$

By differentiating $Q(\Theta, \Theta^{\rm old})$ with respect to ϕ_s, $\theta_{spp_Lp_R}$, and ρ_{srr_L}, the following updating equations are derived:

$$\phi_s^{\rm new} = \frac{1}{G^\phi} \sum_{p,p_L,p_R,r,r_L} \mathscr{C}\big((p,r) \to s(p_L, p_R, r_L); \Theta^{\rm old}\big),$$

$$\theta_{spp_Lp_R}^{\rm new} = \frac{1}{G_{sp}^\theta} \sum_{r,r_L} \mathscr{C}\big((p,r) \to s(p_L, p_R, r_L); \Theta^{\rm old}\big),$$

$$\rho_{srr_L}^{\rm new} = \frac{1}{G_{sr}^\rho} \sum_{p,p_L,p_R} \mathscr{C}\big((p,r) \to s(p_L, p_R, r_L); \Theta^{\rm old}\big)$$

where G's are normalisation constants and we have defined

$$\mathscr{C}\big((p,r) \to s(p_L, p_R, r_L); \Theta^{\rm old}\big) = \sum_{T \in \mathbb{T}_W} \frac{P(T|\Theta^{\rm old}) c((p,r) \to s(p_L, p_R, r_L); T)}{P(W|\Theta^{\rm old})}. \tag{9}$$

($\mathbb{T}_W$ denotes the set of all possible time-span trees of W.) The quantity in Eq. (9) is given as

$$\phi_s^{\rm old} \theta_{spp_Lp_R}^{\rm old} \rho_{srr_L}^{\rm old} \sum_{n=1}^N \sum_{m=n+1}^N \sum_{k=n}^{m-1} \delta_{r_n^m,r} \delta_{r_n^k,r_L} \delta_{r_{k+1}^m, r-r_L} \cdot \alpha_{nmp}^{\rm old}(W) \beta_{nkp_L}^{\rm old}(W) \beta_{(k+1)mp_R}^{\rm old}(W) \tag{10}$$

where $\beta^{\rm old}$ ($\alpha^{\rm old}$) is the inside (outside) variable calculated with $\Theta^{\rm old}$. The case for production rules with parent (S, r_S) is similar.

3.3. Bayesian inference algorithm using Gibbs sampling

Given data W and hyperparameters $\Lambda = (\boldsymbol{\eta}, \boldsymbol{\lambda}_{sp}, \boldsymbol{\nu}_{sr})$, the goal of the Bayesian learning is to estimate the posterior distribution of the model parameters $P(\Theta|W, \Lambda)$, which generally cannot be solved analytically. Fortunately we can develop a Monte-Carlo method to draw samples from $P(T, \Theta|W, \Lambda)$ (T is a time-span tree) via the Gibbs sampling [14].

The Gibbs sampling method is based on alternating samplings of the probabilities $P(\Theta|T, W, \Lambda)$ and $P(T|\Theta, W, \Lambda)$. The former probability can be written as a product of Dirichlet distributions as $P(\boldsymbol{\phi}|T, W, \Lambda) = P_{\rm Dir}(\boldsymbol{\phi}|\boldsymbol{\eta} + \boldsymbol{f}(T))$ where

$$f_s(T) = \sum_{p,p_L,p_R,r,r_L} c\big((p,r) \to s(p_L, p_R, r_L); T\big) \tag{11}$$

and similarly for $\boldsymbol{\lambda}_{sp}$ and $\boldsymbol{\eta}_{sr}$. Therefore Θ can be sampled from $P(\Theta|T, W, \Lambda)$ by sampling from the Dirichlet distributions.

Next, given a set of sampled parameters Θ, a time-span tree T can be sampled from $P(T|\Theta, W, \Lambda) = P(T|\Theta, W)$ by the following recursive sampling for each node. Each node is indicated as a triplet (n, m, p) ($1 \le n \le m \le N$, $p \in \Omega_p$) meaning that the symbol p spans the time span r_n^m. Starting from the top node $(1, N, S)$, if there is a node (n, m, p) with $n<m$, it is expanded by sampling s, k, p_L, p_R from the distribution $P\big((p, r_n^m) \to$

Table 1. Accuracies of time-span tree analyses.

Learning condition	Accuracy (%)
Supervised (open)	44.1
Supervised (closed)	44.9
Unsupervised (EM)	32.3
Unsupervised (Gibbs)	31.4
ATTA	44
σGTTM III	76

Table 2. Results of error analysis.

Height	# nodes	Accuracy (%)	Children matched (%)
1	4237	60.8	80.4
2	2548	43.6	52.7
3	1578	35.0	45.8
4	998	22.8	35.6
5	606	13.2	21.9
6	358	3.9	8.9
7≥	253	3.6	9.1

$s(p_L, r_n^{k-1})(p_R, r_k^m)|\Theta, W\big)$, which is given as

$$\phi_s \theta_{s p p_L p_R} \rho_{s (r_n^m)(r_n^{k-1})} \frac{\beta_{n(k-1)p_L} \beta_{kmp_R}}{\beta_{nmp}} \tag{12}$$

where β's are the inside variables calculated with Θ.

4. EVALUATION

4.1. Evaluation setup

To test PGTTM, we implemented a time-span tree analyser based on the CYK-Viterbi algorithm and evaluated the analyser with a database of manually labelled time-span trees. The database consisted of 300 musical passages with time-span trees analysed by a music expert [9].

To test its ability to learn the parameters, we evaluated the PGTTM in different learning conditions. The first two conditions were supervised learning, one the open condition (piece-wise cross validation) in which the training data did not include the test data and the other the closed condition in which the test data was also used for learning. To avoid zero frequencies, we uniformly added a 0.1 count for all frequencies. The next conditions were unsupervised learning, one based on the EM algorithm (Sec. 3.2) and the other based on the Gibbs sampling method (Sec. 3.3). For the EM algorithm, the initial parameter values were randomly chosen. For the Gibbs sampling method, all hyperparameters were set as 0.1 and after sampling, the parameter set that yielded the maximal likelihood was chosen and applied EM iterations before it was used for the time-span tree analysis.

An estimated time-span tree was compared with the reference in the database and the accuracy was calculated in the following way. First a node of an estimated time-span tree was defined as matched if it had a corresponding node with the same parent and the same children in the reference tree. Then the accuracy was defined as the ratio of the number of matched nodes to the total number of nodes.

4.2. Evaluation results and error analysis

The results in Table 1 show that the accuracies for the PGTTM were nearly equal to that for ATTA. For reference, the accuracy is also shown for σGTTM III, which uses the true group boundary of notes in the annotated data and thus cannot be fairly compared to the other algorithms. The fact that the open and closed tests for supervised learning had nearly equal accuracies indicates that there was little data sparseness and the accuracy would not improve much with a larger data size. The EM algorithm and the Gibbs sampling had similar accuracies that are lower than the case of supervised learning.

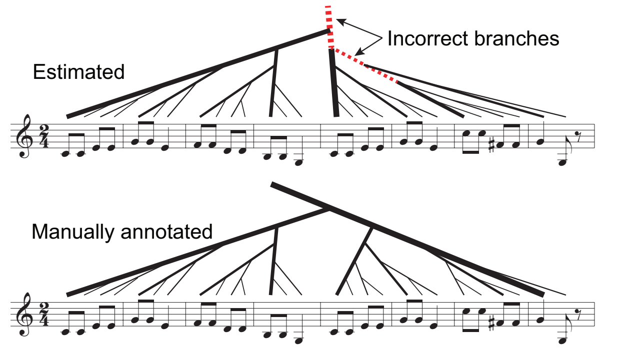

Fig. 2. Example result of time-span tree analysis; an estimated time-span tree (top) and the correct time-span tree (bottom). Incorrectly estimated branches are indicated with red broken lines.

A detailed analysis of accuracies are listed in Table 2 where nodes are classified according to the height, which is defined as the maximal distance from descendant leaves, and the individual accuracies are shown. Because we had similar results, we indicate only the case for the open supervised learning. In the table, the proportion of nodes that has a corresponding node with the same children but not necessarily with the same parent in the reference tree is also shown. The discrepancy between this value and the accuracy indicates the proportion of nodes whose parent was incorrectly estimated. There is a clear tendency that the nodes with lower height have higher accuracies. This can also be confirmed in an example of estimated time-span trees in Fig. 2. The example illustrates the typical case where the cadential note is not correctly selected as the most important note of the passage. Such misidentification of the most important note causes cascading errors in the estimated time-span tree.

4.3. Discussion

It is encouraging that the PGTTM worked as accurately as ATTA without elaborate manual tuning of parameters. However the results also suggest refinements of the model are necessary to improve the accuracy. First, as is the case with NLP, the production rule probabilities depend rather strongly on the context, and it would be important to incorporate dependence between several notes in successions into the model [16]. Another important issue is the chose of symbols in the grammar. Since we have not essentially used non-terminals, the production rules of PGTTM are the same for all positions and heights in the time-span tree. The fact that notes with higher metrical weight are often more important in the lower-height nodes but the clauses are often on weak beats suggests that we need to extend the model with latent symbols, which would improve the time-span tree analysis for nodes with higher heights. Symbol refinement used for NLP [19, 24] can be an aid for this extension.

5. CONCLUSION

Based on GTTM, we have constructed a probabilistic tree structure model of written music. We formulated the PGTTM based on an extension of PCFG, for which both supervised and unsupervised learning techniques can be applied. Despite the conceptually simple construction of the model, the PGTTM produced musical syntactic parsing as accurately as the previously proposed method with elaborate manual tuning of parameters. The results suggest further refinements of the grammatical model. For future work, we plan to apply the model for music transcription and automatic music arrangement and extend the model for polyphony.

6. REFERENCES

[1] A. Klapuri and M. Davy (eds.), *Signal Processing Methods for Music Transcription*, New York: Springer, 2006.

[2] N. Chomsky, *Syntactic Structures*, Mouton & Co., 1957.

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

[4] F. Salzer and C. Schachter, *Counterpoint in Composition*, Columbia University Press, 1969.

[5] S. Kostka, D. Payne and B. Almén, *Tonal harmony* (7th ed.), McGraw-Hill, New York, 2004.

[6] A. Cadwallader and D. Gagné, *Analysis of Tonal Music: A Schenkerian Approach* (3rd ed.), Oxford University Press, 2011.

[7] M. Hamanaka, K. Hirata and S. Tojo, "Implementing 'A Generative Theory of Tonal Music'," *Journal of New Music Research*, vol. 35 no. 4, pp. 249–277, 2006.

[8] Y. Miura, M. Hamanaka, K. Hirata and S. Tojo, "Use of Decision Tree to Detect GTTM Group Boundaries," *Proc. ICMC*, pp. 125–128, 2009.

[9] M. Hamanaka, K. Hirata and S. Tojo, "Music Structural Analysis Database based on GTTM," *Proc. ISMIR*, pp. 325–330, 2014.

[10] M. Hamanaka, K. Hirata and S. Tojo, "σGTTM III: Learning Based Time-Span Tree Generator Based on PCFG," *Proc. CMMR*, pp. 303–317, 2015.

[11] D. Temperley, "A Unified Probabilistic Model for Polyphonic Music Analysis," *Journal of New Music Research*, vol. 38 no. 1, pp. 3–18, 2009.

[12] A. Marsden, "Schenkerian Analysis by Computer: A Proof of Concept," *Journal of New Music Research*, vol. 39 no. 3, pp. 269–289, 2010.

[13] E. Nakamura and S. Sagayama, "Automatic Piano Reduction from Ensemble Scores Based on Merged-Output Hidden Markov Model," *Proc. ICMC*, pp. 298–305, 2015.

[14] M. Johnson, T. Griffiths and S. Goldwater, "Bayesian Inference for PCFGs via Markov Chain Monte Carlo," *Proc. HLT-NAACL*, pp. 139–146, 2007.

[15] M. Post and D. Gildea, "Bayesian Learning of a Tree Substitution Grammar," *Proc. ACL-IJCNLP*, pp. 45–48, 2009.

[16] T. Cohn, P. Blunsom and S. Goldwater, "Inducing Tree-Substitution Grammars," *Journal of Machine Learning Research*, vol. 11, pp. 3053–3096, 2010.

[17] H. Shindo, Y. Miyao, A. Fujino and M. Nagata, "Bayesian Symbol-Refined Tree Substitution Grammars for Syntactic Parsing," *Proc. ACL*, pp. 440–448, 2012.

[18] C. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*, MIT Press, Cambridge, 1999.

[19] M. Johnson, "PCFG Models of Linguistic Tree Representations," *Computational Linguistics*, **24**, pp. 613–632, 1998.

[20] M. Nakano, Y. Ohishi, H. Kameoka, R. Mukai and K. Kashino, "Bayesian Nonparametric Music Parser," *Proc. ICASSP*, pp. 461–464, 2012.

[21] W. Granroth and M. Steedman, "Statistical Parsing for Harmonic Analysis of Jazz Chord Sequences," *Proc. ICMC*, pp. 478–485, 2012.

[22] D. Temperley, *Music and Probability*, MIT Press, 2006.

[23] R. Neal and G. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models*, Springer Netherlands, pp. 355–368, 1998.

[24] T. Matsuzaki, Y. Miyao and J. Tsujii, "Probabilistic CFG with Latent Annotations," *Proc. ACL*, pp. 75–82, 2005.