FUNCTION- AND RHYTHM-AWARE MELODY HARMONIZATION BASED ON TREE-STRUCTURED PARSING AND SPLIT-MERGE SAMPLING OF CHORD SEQUENCES

Hiroaki Tsushima[1] **Eita Nakamura**[1] **Katsutoshi Itoyama**[1] **Kazuyoshi Yoshii**[1,2]
[1]Graduate School of Informatics, Kyoto University, Japan [2]RIKEN AIP, Japan
{tsushima, enakamura}@sap.ist.i.kyoto-u.ac.jp, {itoyama, yoshii}@kuis.kyoto-u.ac.jp

ABSTRACT

This paper presents an automatic harmonization method that, for a given melody (sequence of musical notes), generates a sequence of chord symbols in the style of existing data. A typical way is to use hidden Markov models (HMMs) that represent chord transitions on a regular grid (e.g., bar or beat grid). This approach, however, cannot explicitly describe the rhythms, harmonic functions (e.g., tonic, dominant, and subdominant), and the hierarchical structure of chords, which are supposedly important in traditional harmony theories. To solve this, we formulate a hierarchical generative model consisting of (1) a probabilistic context-free grammar (PCFG) for chords incorporating their syntactic functions, (2) a metrical Markov model describing chord rhythms, and (3) a Markov model generating melodies conditionally on a chord sequence. To estimate a variable-length chord sequence for a given melody, we iteratively refine the latent tree structure and the chord symbols and rhythms using a Metropolis-Hastings sampler with split-merge operations. Experimental results show that the proposed method outperformed the HMM-based method in terms of predictive abilities.

1. INTRODUCTION

Creation of chord sequences plays a key role in music composition and arrangement since harmony affects the mood of music and characterizes the impression of a certain musical style. Our aim is automatic melody harmonization, or automatic generation of a sequence of chord symbols for a given melody (a sequence of musical notes). In this paper, we restricted our focus to the automatic harmonization in the style of popular music. Instead of manually describing music theories for the style such as jazz and classical music, we take a statistical approach to automatically learn model architectures and parameters from a music corpus and harmonize in the style of that data. We formulate a probabilistic model that represents how likely a chord sequence is to be generated and another model that represents how likely a melody is to be generated conditionally on a chord sequence.

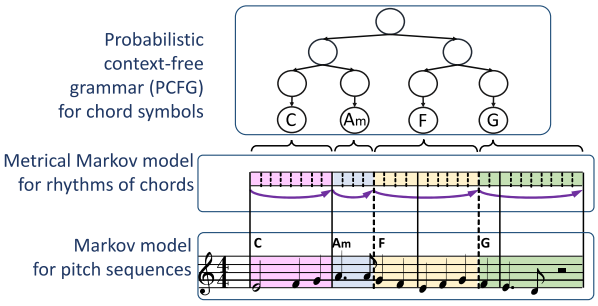

Figure 1: The hierarchical generative model for chord symbol, chord rhythms, and melodies.

Since chord sequences are usually described by Markov models [21, 25], a standard way to statistical harmonization is to use a hidden Markov model (HMM) that has a latent Markov chain of chord symbols and assumes a musical note sequence to be generated conditionally on the chords. This approach, however, does not consider the syntactic roles and hierarchical structure of chords. In traditional harmony theories (e.g., [14, 22]), such syntactic roles are often referred to as *harmonic functions* (e.g., tonic, dominant, and subdominant), which are similar to parts-of-speech in language theories. Another problem of the conventional HMMs lies in the description of the chord rhythms (onset score times or durations of chords). Since chord durations are described by self-transition probabilities on a regular time grid (e.g., beat or bar grid), the chord rhythms are not explicitly described.

To solve these problems, we propose a tree-structured hierarchical generative model that consists of (1) a probabilistic context-free grammar (PCFG) that generates chord symbols, (2) a metrical Markov model that generates chord rhythms, and (3) a Markov model that generates a melody from a chord sequence (Fig. 1). The use of the PCFG was inspired by Steedman's pioneering work [27] that uses a context-free grammar (CFG) for representing the hierarchical structure of chords. A key advantage of our study is that the rule probabilities and tree structure of the PCFG can jointly be estimated in an unsupervised manner from a corpus of chord sequences, expecting that the syntactic roles of chords are captured by the non-terminal symbols.

The metrical Markov model is used for explicitly describing transition probabilities between the onset beat positions of succeeding chords.

Using the tree-structured hierarchical generative model, we propose a statistical harmonization method based on a sophisticated Metropolis-Hasting (MH) sampler with split-merge operations. To estimate a variable-length chord sequence with appropriate chord rhythms for a given melody, we stochastically search for the most likely latent tree structure, symbols, and onset score times of chords from their posterior distributions. In this search, our sampler has four types of proposals: the whole latent tree structure is updated using a variant of the Viterbi algorithm, one of the chords is split or two adjacent chords are merged according to the latent tree structure, and one of the chord onset score time is moved back or forth. Such stochastic global or local updates can be interpreted as a repeated trial-and-error process of finding an optimal chord sequence.

2. RELATED WORK

This section introduces related studies on the automatic harmonization and on the music language model for chords and notes.

2.1 Automatic Harmonization

Some studies on harmonization aim to generate sequences of chord symbols (as in this paper) and other studies aim to generate several (typically four) voices of musical notes. In the former direction, Chuan and Chew [3] proposed a hybrid method that consists of three processes: selection of chord tones (constituent tones of chords) from given melodies with a support vector machine (SVM), construction of triad chords from chord tones, and generation of chord progressions by a rule-base method. Simon et al. [25] developed a method based on HMMs in which chord transitions are described by Markov models. This method has been implemented in a commercial system *MySong*. Raczyński et al. [21] proposed similar Markov models in which chords are conditioned by melodies and time-varying keys. To our knowledge, PCFG has not been used for melody harmonization.

In the latter direction, Ebcioğlu [5] proposed a rule-based method for generating four-part chorales in Bach's style. Methods by using variants of genetic algorithms (GAs) based on music theories have also been studied [18, 19, 28]. Allan and Williams et al. [1] proposed a method based on HMMs which represent chords as hidden states and musical notes as observed outputs. A hidden semi-Markov model (HSMM) [9] has been used for explicitly representing the durations of chords. Paiement et al. [17] proposed a hierarchical tree-structured model that describes chord movements from the viewpoint of hierarchical time scales by dividing the notations of chords.

2.2 Music Language Modeling

Several language models for musical notes have been studied for music structure analysis [10, 11, 13, 15]. According to the *Generative Theory of Tonal Music* (GTTM) [13], a note sequence is assumed to have a hierarchical structure that describes which notes are important. This theory consists of rules for recursively reducing a note sequence into a single note. Computational implementation of GTTM and the analysis of musical pieces using it have been studied [10, 11]. A probabilistic formulation of GTTM based on PCFG has been proposed and enabled unsupervised learning of production rules directly from note sequences [15].

Various language models for chord sequences have been proposed in the context of automatic chord recognition for music audio signals [16, 24, 29], music analysis [23, 27], and music arrangement [6, 20]. The conventional language model for chord sequences is n-gram models [6,24]. To avoid the sparseness problem with a large value of n, smoothing methods have been studied for improving the predictive ability of the language model [2]. Yoshii et al. [29] proposed a vocabulary-free infinity-gram model in which each chord depends on a variable-length history of chords. Paiement et al. [16] introduced several hidden layers of state transitions that represent the hierarchical structure of chords. Some studies attempted to explicitly describe the generative grammar to represent the hierarchical structure of chords [20, 23, 27]. Steedman [27] and Rohrmeier [23] proposed a description of the production rules for chord sequences. A probabilistic extension is later studied in the context of music arrangement and unsupervised learning of the probabilities has been performed [20]. In these studies, the lists of non-terminals and production rules were manually given based on music theories or musical intuition.

3. PROBABILISTIC MODELING

This section explains how to formulate and train our hierarchical generative model of chords and melodies. In our model, the PCFG for chord symbols is trained by unsupervised learning from a corpus of chord sequences while estimating the latent tree structures behind these sequences. The metrical Markov model for chord rhythms is trained by supervised learning from a corpus including chord rhythms. The Markov model for pitch sequences is also trained by supervised learning from paired data of melodies and chord sequences.

3.1 Model Formulation

The PCFG stochastically generates a sequence of chord symbols (or simply *chords* in the following) $\boldsymbol{z} = \{z_n\}_{n=1}^N$ and the metrical Markov model generates the corresponding onset score times $\boldsymbol{\phi} = \{\phi_n\}_{n=1}^N$ described in units of 16th notes, where N is the number of chords. A subsequence of pitches in the melody $\boldsymbol{x}_n = \{x_{n,i}\}_{i=1}^{I_n}$ in the time span of each chord z_n is then generated, where I_n is the number of pitches in that time span. Concatenating all such subsequences, the whole sequence of pitches $\boldsymbol{x} = \{\boldsymbol{x}_n\}_{n=1}^N$ is obtained. $I = \sum_{n=1}^N I_n$ denotes the number of melody notes. Let $\psi_{n,i}$ be the onset score time of the melody note corresponding to $x_{n,i}$ and let $\boldsymbol{\psi} = \{\{\psi_{n,i}\}_{i=1}^{I_n}\}_{n=1}^N$. ϕ_n and $\psi_{n,i}$ can take integer values from 0 to $16L-1$, where L is the total number of measures in the whole melody. Although in the training phase, we have multiple sequences of chords sequences

and melodies, we formulate here the case of a single sequence for notational simplicity. Extension for multiple sequences is straightforward.

The PCFG G is defined by

$$G = (V, \Sigma, R, S), \tag{1}$$

where V is a set of non-terminal symbols, which are expected to represent hierarchical structure and syntactic roles of chords, Σ is a set of terminal symbols (chord symbols), R is a set of rule probabilities, and S is a start symbol (a non-terminal symbol located on the root of a syntax tree). Rule probabilities consist of the following three types. $\theta_{A\to BC}$ is the probability that a non-terminal symbol $A \in V$ branches to non-terminal symbols $B \in V$ and $C \in V$. $\eta_{A\to\alpha}$ is the probability that $A \in V$ emits terminal symbol $\alpha \in \Sigma$. Each non-terminal symbol $A \in V$ has a coin-toss probability λ_A that stochastically determines whether A emits (otherwise A branches). These probabilities should be normalized properly as follows:

$$\sum_{B,C\in V} \theta_{A\to BC} = 1, \quad \sum_{\alpha\in\Sigma} \eta_{A\to\alpha} = 1. \tag{2}$$

We define $\boldsymbol{\theta}_A = \{\theta_{A\to BC}\}_{B,C\in V}$, $\boldsymbol{\eta}_A = \{\eta_{A\to\alpha}\}_{\alpha\in\Sigma}$, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_A\}_{A\in V}$, $\boldsymbol{\eta} = \{\boldsymbol{\eta}_A\}_{A\in V}$, etc. Similar notations are used throughout this paper.

The metrical Markov model describes the transition probabilities for chord onset beat positions (16th-note level relative score time in a measure) as

$$p(\phi_n|\phi_{n-1}) = \pi_{\bar{\phi}_{n-1},\bar{\phi}_n}, \tag{3}$$

where $\bar{\phi}_n = \phi_n \bmod 16$ and π_{ab} $(0 \le a, b < 16)$ indicates the transition probability from beat position k to beat position l. When $\bar{\phi}_n \le \bar{\phi}_{n-1}$, we interpret that the onset of chord n is in the next measure.

The Markov model is described with the following transition probability:

$$p(x_{n,m}|x_{n,m-1}, z_n) = \tau^{z_n}_{x_{n,m-1},x_{n,m}}, \tag{4}$$

where $\tau^{z_n}_{x_{n,m-1},x_{n,m}}$ is the transition probability from pitch $x_{n,m-1}$ to pitch $x_{n,m}$ under chord z_n. In addition, the probability of the first pitch in $\boldsymbol{x}_n$ is given by $p(x_{n,1}|z_n)$.

We put conjugate priors on the parameters of the PCFG as

$$\boldsymbol{\theta}_A \sim \text{Dirichlet}(\boldsymbol{\xi}_A), \tag{5}$$

$$\boldsymbol{\eta}_A \sim \text{Dirichlet}(\boldsymbol{\zeta}_A), \tag{6}$$

$$\lambda_A \sim \text{Beta}(\boldsymbol{\iota}_A), \tag{7}$$

where $\boldsymbol{\xi}_A$, $\boldsymbol{\zeta}_A$ and $\boldsymbol{\iota}_A$ are hyperparameters. Similarly, we put conjugate priors on the parameters of the Markov models as follows:

$$\boldsymbol{\pi}_a \sim \text{Dirichlet}(\boldsymbol{\beta}), \tag{8}$$

$$\boldsymbol{\tau}^z_x \sim \text{Dirichlet}(\boldsymbol{\gamma}), \tag{9}$$

where $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are hyperparameters.

To complete the generative model of chords and melodies, we need to specify a model generating $\boldsymbol{\psi}$. This model can be formulated as for the model of $\boldsymbol{\phi}$ and we omit the details here since melodies are given as inputs for our harmonization problem.

3.2 Bayesian Learning

We obtain the model parameters $\boldsymbol{\Theta} = \{\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\lambda}, \boldsymbol{\pi}, \boldsymbol{\tau}\}$ by the maximum posterior (MAP) estimation. To estimate the parameters $\boldsymbol{\theta}$, $\boldsymbol{\eta}$, and $\boldsymbol{\lambda}$ of the PCFG, we use a variant of Gibbs sampling called the inside-filtering-outside-sampling algorithm [12]. We assume that a chord sequence $\boldsymbol{z}$ was derived from a latent syntactic tree $\boldsymbol{t}$. $\boldsymbol{t}$ can be represented by a set of non-terminal nodes $\{t_{n:m}\}_{1\le n\le m\le N}$, where $t_{n:m}$ is the root node of a subtree that derives a subsequence of chords $z_{n:m} = \{z_n, z_{n+1}, \cdots, z_m\}$. The latent tree $\boldsymbol{t}$ and the parameters $\boldsymbol{\theta}$, $\boldsymbol{\eta}$, and $\boldsymbol{\lambda}$ are alternately sampled from the *conditional* posterior distributions $p(\boldsymbol{t}|\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\lambda}, \boldsymbol{z})$ and $p(\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\lambda}|\boldsymbol{t}, \boldsymbol{z})$. This algorithm is proven to yield samples of $\boldsymbol{t}$, $\boldsymbol{\theta}$, $\boldsymbol{\eta}$, $\boldsymbol{\lambda}$ following the true posterior distribution $p(\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\lambda}, \boldsymbol{t}|\boldsymbol{z})$.

In the inside filtering step, we focus on the conditional probability (*inside probability*) that a subsequence $z_{n:m}$ is derived from a subtree whose root node is A

$$p^A_{n,m} = p(z_{n:m}|t_{n:m} = A). \tag{10}$$

This probability can be calculated recursively from the leaf nodes to the root node as follows:

$$p^A_{n,n} = \lambda_A \eta_{A\to z_n}, \tag{11}$$

$$p^A_{n,n+k} = \sum_{B,C\in V} \left[(1-\lambda_A)\theta_{A\to BC} \sum_{1\le l\le k} p^B_{n,n+l-1} p^C_{n+l,n+k} \right].$$

In the outside sampling step, we recursively sample a latent tree $\boldsymbol{t}$ from a start symbol S to the leaf nodes according to $p(\boldsymbol{t}|\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\lambda}, \boldsymbol{z})$ by using the inside probabilities. When a node $t_{n:n+k} = A$ is already sampled, the two non-terminal symbols B and C into which $t_{n:n+k}$ branches are sampled as follows:

$$\begin{aligned} &p(l, B, C) \\ &= p(t_{n:n+l-1} = B, t_{n+l:n+k} = C \,|\, t_{n:n+k} = A, z_{n:n+k}) \\ &= (1-\lambda_A)\theta_{A\to BC}\, p^B_{n,n+l-1}\, p^C_{n+l,n+k} / p^A_{n,n+k}, \end{aligned} \tag{12}$$

where $1 \le l \le k$ indicates a split position.

Finally, we sample parameters $\boldsymbol{\theta}$, $\boldsymbol{\eta}$, and $\boldsymbol{\lambda}$ according to $p(\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\lambda}|\boldsymbol{t}, \boldsymbol{z}) = p(\boldsymbol{\theta}|\boldsymbol{t}, \boldsymbol{z}) p(\boldsymbol{\eta}|\boldsymbol{t}, \boldsymbol{z}) p(\boldsymbol{\lambda}|\boldsymbol{t}, \boldsymbol{z})$ given by

$$\boldsymbol{\theta}_A \sim \text{Dirichlet}(\boldsymbol{\xi}_A + \boldsymbol{u}_A), \tag{13}$$

$$\boldsymbol{\eta}_A \sim \text{Dirichlet}(\boldsymbol{\zeta}_A + \boldsymbol{v}_A), \tag{14}$$

$$\lambda_A \sim \text{Beta}(\boldsymbol{\iota}_A + \boldsymbol{w}_A), \tag{15}$$

where $u_{A\to BC}$ $(v_{A\to\alpha})$ is the number of times the binary production rule $\theta_{A\to BC}$ (the emission rule $\eta_{A\to\alpha}$) is used in $\boldsymbol{t}$, and $w_{A,0}$ $(w_{A,1})$ is the number of times a non-terminal symbol A branches (emits) and $\boldsymbol{t}$.

The parameters $\boldsymbol{\pi}$ and $\boldsymbol{\tau}$ of the Markov models are obtained by supervised learning. Given $\boldsymbol{\phi}$, the posterior distribution of $\boldsymbol{\pi}$ can be calculated easily because of the conjugacy between the Dirichlet and categorical distributions. Similarly, given paired data of $\boldsymbol{z}$, $\boldsymbol{\phi}$, $\boldsymbol{x}$, and $\boldsymbol{\psi}$, the posterior distribution of $\boldsymbol{\tau}$ can be calculated.

4. AUTOMATIC HARMONIZATION

This section explains how to generate sequences of chords for a given melody by using the model in Section 3.

4.1 Problem Specification

Given a melody with pitches $\boldsymbol{x}$ and onset score times $\boldsymbol{\psi}$ and trained model parameters $\boldsymbol{\Theta}$, we aim to estimate a variable-length sequence of chords $\boldsymbol{z}$ and their onset score times $\boldsymbol{\phi}$ that are not restricted to bar lines. Note that the number of chords N is not fixed and should be estimated and that a latent tree $\boldsymbol{t}$ for chords is considered and estimated unlike conventional harmonization methods.

4.2 Metropolis-Hastings Sampling

We propose a Metropolis-Hastings (MH) sampler with split-merge operations for generating samples of $\boldsymbol{t}$, $\boldsymbol{z}$, and $\boldsymbol{\phi}$ from the posterior distribution $p(\boldsymbol{t}, \boldsymbol{z}, \boldsymbol{\phi}|\boldsymbol{x}, \boldsymbol{\psi}, \boldsymbol{\Theta})$ based on the following four types of proposals:

- **Global update:** Update chords $\boldsymbol{z}$ and latent tree $\boldsymbol{t}$ with a Viterbi algorithm for the PCFG, keeping the number and score times of chords unchanged.
- **Split operation:** Randomly choose one of the chords and split it into two adjacent chords.
- **Merge operation:** Randomly choose two adjacent chords in $\boldsymbol{z}$ that form a subtree with two leaves in $\boldsymbol{t}$ and merge them.
- **Rhythm update:** Randomly choose one chord n and move its onset score time ϕ_n back or forth.

Although it is more proper to use the inside-filtering-outside-sampling algorithm for the global update, the Viterbi algorithm is used for efficient optimization in the posterior space.

In the MH sampling, one of these proposals is randomly selected. More specifically, the sampler proposes a sample $s^* = (\boldsymbol{t}, \boldsymbol{z}, \boldsymbol{\phi})^*$ from a current sample $s = (\boldsymbol{t}, \boldsymbol{z}, \boldsymbol{\phi})$. The sampler then judges whether s^* is accepted as the next sample or not according to the acceptance ratio given by

$$g(s^*, s) = \min\left\{1, \frac{p(s^*)p(s|s^*)}{p(s)p(s^*|s)}\right\}, \tag{16}$$

where $p(s)$ is the complete joint likelihood of s based on the proposed model and $p(s^*|s)$ is a proposal distribution that should be set appropriately. If the proposal is rejected, $\boldsymbol{t}$, $\boldsymbol{z}$, and $\boldsymbol{\phi}$ are not updated. In our method, there are three types of proposal distributions for the second to fourth proposals in the above list. We estimate the most plausible $\mathbf{z}$ and ϕ by iterating the MH sampling a sufficient number of times and then getting the latent variables that maximize the likelihood of complete data.

4.3 Updating Chord Symbols

We describe how to update the chord symbols $\boldsymbol{z}$ and the corresponding latent tree $\boldsymbol{t}$ according to the conditional posterior distribution $p(\boldsymbol{t}, \boldsymbol{z}|\boldsymbol{\phi}, \boldsymbol{x}, \boldsymbol{\psi}, \boldsymbol{\Theta})$.

4.3.1 Viterbi Algorithm

Given a melody with pitches $\boldsymbol{x}$ and onset score times $\boldsymbol{\psi}$, we can efficiently sample a sequence of chord symbols $\boldsymbol{z}$ and the corresponding latent tree $\boldsymbol{t}$ by using the Viterbi algorithm. Our algorithm differs from a standard Viterbi algorithm used for estimating $\boldsymbol{t}$ for a given $\boldsymbol{z}$ because both $\boldsymbol{t}$ and $\boldsymbol{z}$ are the latent variables to be estimated in this paper.

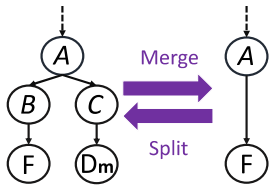

Figure 2: The split-merge operations of the MH sampling.

We first recursively calculate the inside probabilities from the layer of terminal symbols $\boldsymbol{z}$ to the start symbol S according to

$$p^A_{n,n} = \lambda_A \max_{c\in\Sigma} \eta_{A\to c}\, p(\boldsymbol{x}_n|c)^{1/I_n}, \tag{17}$$
$$p^A_{n,n+k} = (1-\lambda_A) \max_{\substack{B,C\in V\\ 1\le l\le k}} \theta_{A\to BC} p^B_{n,n+l-1} p^C_{n+l,n+k},$$

where $p(\boldsymbol{x}_n|c)$ is the probability that a pitch subsequence $\boldsymbol{x}_n$ is generated conditionally on a chord c:

$$p(\boldsymbol{x}_n|c) = p(x_{n,1}|c)\prod_{i=2}^{I_n} p(x_{n,i}|x_{n,i-1}, c). \tag{18}$$

The most likely $\boldsymbol{t}$ and $\boldsymbol{z}$ are obtained by recursively backtracking the most likely paths from the start symbol S.

4.3.2 Split-Merge Operations

Using the MH sampler, we can split a chord or merge adjacent chords by considering the underlying tree $\boldsymbol{t}$ and the emission probability of the melody. Note the split and merge operations are inverse to each other and that the latent tree $\boldsymbol{t}$ is locally updated by these operations (Fig. 2).

In the split operation, a new sample s^* is proposed by stochastically selecting a chord z_n from $\boldsymbol{z}$, splitting z_n into z^L and z^R, selecting the new onset score time $\phi^* \in (\phi_n, \phi_{n+1}) = [\phi_n + 1, \phi_{n+1} - 1]$, and splitting the non-terminal symbol $t_{n:n}$ into two non-terminal symbols t^L and t^R. The proposal distribution $p(s^*|s)$ is thus

$$p(s^*|s) = \begin{cases} \frac{\theta_{t_{n:n}\to t^L t^R}\eta_{t^L\to z^L}\eta_{t^R\to z^R}}{N(\phi_{n+1}-\phi_n-1)}, & \phi_{n+1} \ge \phi_n + 1; \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

The reverse proposal distribution $p(s|s^*)$, on the other hand, is same as the proposal distribution for the merge operation in which a sample s is proposed by stochastically selecting a pair of adjacent chords z^L and z^R, merging those chords into z_n, by selecting a chord z_n according to the probability $\eta_{t_{n:n}\to z_n}$. Thus we have

$$p(s|s^*) = \frac{\eta_{t_{n:n}\to z_n}}{\#\text{MergeableNodes}(s^*)}, \tag{20}$$

where $\#\text{MergeableNodes}(s^*)$ is the number of pairs of adjacent chords that can be merged in s^*, i.e., those chords forming a subtree with two leaves.

The likelihood ratio of $p(s^*)$ to $p(s)$ is then given by

$$\frac{p(s^*)}{p(s)} = \frac{(1-\lambda_{t_{n:n}})\theta_{t_{n:n}\to t^L t^R}\lambda_{t^L}\eta_{t^L\to z^L}\lambda_{t^R}\eta_{t^R\to z^R}}{\lambda_{t_{n:n}}\,\eta_{t_{n:n}\to z_n}} \cdot \frac{p(\boldsymbol{x}^L|z^L)p(\boldsymbol{x}^R|z^R)p(\phi^*|\phi_n)p(\phi_{n+1}|\phi^*)}{p(\boldsymbol{x}_n|z_n)p(\phi_{n+1}|\phi_n)}, \tag{21}$$

where $\boldsymbol{x}^L$ and $\boldsymbol{x}^R$ are the subsequences of pitches obtained by splitting $\boldsymbol{x}_n$ at the score time ϕ^*. Using Eqs. (19), (20), and (21), we can calculate the acceptance ratio of s^* according to Eq. (16).

In the merge operation, on the other hand, a new sample s^* is proposed in a similar way to the split operation. More specifically, the acceptance ratio of s^* given by Eq. (16) can be calculate by exchanging s and s^* in Eqs. (19), (20), and (21). Through the split-merge operations, the number of chords N is optimized stochastically.

4.4 Updating Chord Rhythms

We describe how to update the chord rhythms $\boldsymbol{\phi}$ according to the conditional posterior distribution $p(\boldsymbol{\phi}|\boldsymbol{t},\boldsymbol{z},\boldsymbol{x},\boldsymbol{\psi},\boldsymbol{\Theta})$. A new sample s^* is proposed by stochastically selecting a chord n and moving ϕ_n to a new score time $\phi_n^* \in (\phi_{n-1},\phi_{n+1})$. The proposal distribution $p(s^*|s)$ and the reverse proposal distribution $p(s|s^*)$ are given by

$$p(s^*|s) = p(s|s^*) = \frac{1}{N-1}\frac{1}{\phi_{n+1}-\phi_{n-1}-1}. \tag{22}$$

The likelihood ratio of $p(s^*)$ to $p(s)$ is given by

$$\frac{p(s^*)}{p(s)} = \frac{p(\boldsymbol{x}^*_{n-1}|z_{n-1})p(\boldsymbol{x}^*_n|z_n)p(\phi^*_n|\phi_{n-1})q(\phi_{n+1}|\phi^*_n)}{p(\boldsymbol{x}_{n-1}|z_{n-1})p(\boldsymbol{x}_n|z_n)p(\phi_n|\phi_{n-1})p(\phi_{n+1}|\phi_n)}, \tag{23}$$

where $\boldsymbol{x}^*_{n-1}$ and $\boldsymbol{x}^*_n$ are the subsequences of pitches in the time spans of chords $n-1$ and n with the new onset score time ϕ^*_n. Using Eqs. (22) and (23), we can calculate the acceptance ratio of s^* according to Eq. (16).

5. EVALUATION

In this section, we report two experiments conducted to quantitatively evaluate the proposed generative model and the proposed method of automatic harmonization based on the model and discuss examples of chord sequences generated by the method.

5.1 Experimental Conditions

To learn the PCFG unsupervisedly, we extracted 1002 chord sequences corresponding to sections (e.g., verse, bridge, and chorus) from 468 pieces of popular music included in the SALAMI dataset [26]. Only those sequences with a length between 8 and 32 were chosen. The vocabulary of chord symbols was given by the combinations of 12 root notes {C, C#, D, ..., B} and 2 chord types {major, minor}, and a special "other". The values of the hyperparameters were all set to 0.1.

To train the two Markov models in a supervised manner, we extracted 9902 pairs of melodies and the corresponding chord sequences from 194 pieces of popular music included in Rock Corpus [4]. The values of the hyperparameters were all set to 0.1.

In the testing phase, we extracted 339 pairs of melodies and the corresponding chord sequences as ground-truth data for evaluation from 69 pieces of popular music included in the RWC music database [7, 8]. Note that all the data (SALAMI, Rock Corpus, and RWC) were transposed to C major or C minor.

5.2 Evaluating Ability of Melody Prediction

To evaluate the hierarchical generative model based on the PCFG in terms of the ability of melody prediction, we calculated the marginal likelihood for the melodies extracted from the RWC music database. The number of kinds of non-terminal symbols, or the complexity of the PCFG, K was changed from 1 to 20. In each of cases for K, we obtained different PCFG's parameters with Gibbs sampling and calculated the marginal likelihood for each parameter set. The number of different parameter sets were between 37 and 50 depending on the computational complexity. We assumed that the chord onsets were completely synchronized with bar lines such that the chord sequences were marginalized analytically. The proposed model was compared with an HMM that learns the chord-symbol transition between adjacent units which were either musical notes or measures. When minimum time units were musical notes, each note was assumed to be generated conditionally on the chord symbol at the time. When minimum time units were musical measures, notes accompanying each chord were assumed to be generated according to the probability described in Eq. (4).

The marginal likelihood of the trained model parameters $\boldsymbol{\Theta}$ for an unseen melody $\boldsymbol{X}$ with $\boldsymbol{\psi}$ can be calculated with the inside algorithm in Section 3.2. To sum over all possibilities of a latent chord sequence $\boldsymbol{Z}$ and a latent tree $\boldsymbol{T}$, $p^A_{i,i}$ in Eq. (11) is replaced with

$$p^A_{i,i} = \lambda_A \sum_{c\in\Sigma} \eta_{A\to c}\, p(X_i|c), \tag{24}$$

where $p(X_i|c)$ is given by Eq. (18). The average marginal likelihood $\mathcal{L}$ per note for the melody is given by

$$\mathcal{L} = \frac{1}{I}\log p(\boldsymbol{X}|\boldsymbol{\psi},\boldsymbol{\Theta}) = \frac{1}{I}\log p^S_{0,N-1}, \tag{25}$$

where I (N) is the number of notes (chords).

The experimental results are shown in Fig. 3. The proposed model outperformed the HMM, whether the minimum time unit is a musical note ($\mathcal{L} = -3.2813$) or a measure ($\mathcal{L} = -2.3218$). The likelihood tended to decrease as the value of K increased to eight, and the likelihood tended to increase as the value of K increased beyond eight.

5.3 Evaluating Predictive Ability of Chord Sequences

To evaluate the proposed harmonization method in terms of the predictive ability of unseen chord sequences, we generated chord sequences for the melodies of the RWC

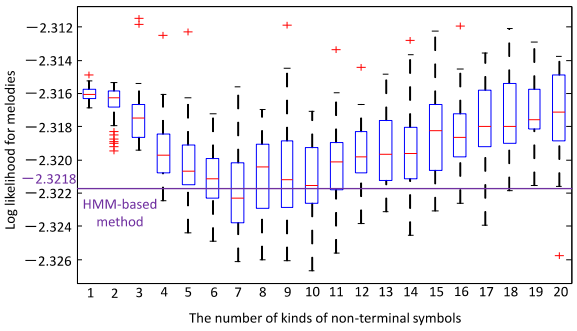

Figure 3: Marginal likelihood for melodies per note. In the box plots, the red line, the black cross and the red crosses indicate the median, the mean and outliers, respectively.

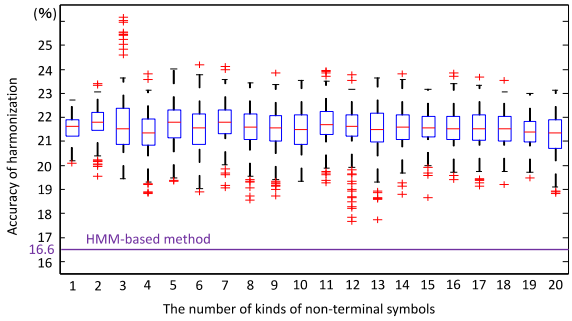

Figure 4: Accuracy of harmonization per note. Indicators in the box plots are the same as those in Fig. 3

music database and calculated the accuracy at a 16th-note level compared with the ground-truth. The complexity of the PCFG, K was changed from 1 to 20. The proposed method was compared with a conventional HMM-based method that represents chord transitions on a 16th-note-level grid.

The experimental result are shown in Fig. 4. The proposed model clearly outperformed the HMM-based method with an accuracy of 16.6 %. While a certain range of K showed much higher accuracy (e.g., 26 %) than the HMM-based method, there was little correlation between K and the median values of accuracies.

5.4 Generated Example and Discussion

Fig. 5 shows how the proposed MH sampling method with split-merge operations worked for automatic harmonization[1] . The number of kinds of non-terminal symbols, K, was set to 12. The chord sequence at the top shows an initial sample in which the chord symbols were optimized by the Viterbi algorithm, but the chord onsets were located at the bar lines. The second chord sequence shows a sample proposed by moving the onset positions of 5th and 6th chords (G major and C major). The third chord sequence shows a sample proposed by merging the 7th and 8th chords (F major and C major) into one chord (C major). The bottom chord sequence shows the best sample that maximizes the likelihood for the given melody. In each of the processes, the likelihood increased. The result indicates that the proposed method can successfully generate a variable-length sequence of chords by considering the latent tree structure behind the chord sequence.

[1] Some chord sequences generated in this experimental are available online: http://sap.ist.i.kyoto-u.ac.jp/members/tsushima/ismir2017/

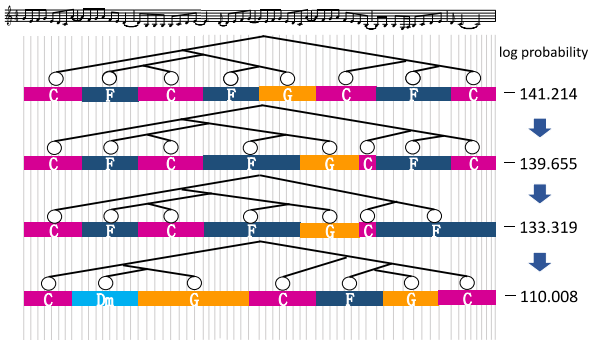

Figure 5: Sampling-based estimation of the most likely chord sequence for a given melody.

We found some problems to be tackled in the future. The proposed method tended to generate simple chords (e.g., C major and A minor). This is because the chord symbols were refined by using the Viterbi algorithm. In addition, the number of the most plausible chord sequences selected in our experiment was rarely more than those initialized at the beginning of sampling. This is because the proposals of the split operation were accepted less frequently than the proposals of the merge operation.

6. CONCLUSION

This paper presented an automatic harmonization method that generates a variable-length chord sequence for a given melody based on music rules hidden in corpora of popular music. The experimental results showed that the proposed model outperformed the HMM-based method in terms of predictive ability and has a large potential for statistical music composition or arrangement.

Since our method is based on statistical learning, it was found to prefer simpler and basic chord sequences. More specifically, the number of generated chords tends to be less than the number of measures. This problem could be solved by giving more chances to the split operation in MCMC sampling. To increase the diversity of generated chord symbols, a sampling or beam-search method is considered to be effective instead of the Viterbi algorithm that tends to find a popular chord sequence that has the highest posterior probability from the statistical viewpoint.

We still need further studies on our model. In this paper, one measure with the time signature of 4/4 is divided into 16 time units. It is therefore important to investigate the best time resolution and extend the model to deal with other kinds of time signatures. In addition, to evaluate the musical appropriateness of generated chord sequences, we plan to conduct a subjective listening test and evaluate how consistent our model is with music theories or musical intuition.

Acknowledgement: This study was partially supported by JSPS KAKENHI Grant Numbers 26700020, 16H01744, and 16J05486 and JST ACCEL No. JPMJAC1602.

7. REFERENCES

[1] M. Allan and C. Williams. Harmonising chorales by probabilistic inference. In *Advances in Neural Information Processing Systems (NIPS)*, pages 25–32, 2005.

[2] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Association for Computational Linguistics (ACL)*, pages 310–318, 1996.

[3] C. H. Chuan and E. Chew. A hybrid system for automatic generation of style-specific accompaniment. In *Proceedings of the 4th International Joint Workshop on Computational Creativity*, pages 57–64, 2007.

[4] T. D. Clercq and D. Temperley. A corpus analysis of rock harmony. *Popular Music*, 30(01):47–70, 2011.

[5] K. Ebcioğlu. An expert system for harmonizing four-part chorales. *Computer Music Journal*, 12(3):43–51, 1988.

[6] S. Fukayama, K. Yoshii, and M. Goto. Chord-sequence-factory: A chord arrangement system modifying factorized chord sequence probabilities. In *ISMIR*, 2013.

[7] M. Goto. Aist annotation for the RWC music database. In *ISMIR*, pages 359–360, 2006.

[8] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical and jazz music databases. In *ISMIR*, volume 2, pages 287–288, 2002.

[9] R. Groves. Automatic harmonization using a hidden semi-Markov model. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference (Boston, MA)*, pages 48–54, 2013.

[10] M. Hamanaka, K. Hirata, and S. Tojo. Implementing 'A generative theory of tonal music'. *Journal of New Music Research*, 35(4):249–277, 2006.

[11] M. Hamanaka, K. Hirata, and S. Tojo. Musical structural analysis database based on GTTM. In *ISMIR*, pages 325–330, 2014.

[12] M. Johnson, T. L. Griffiths, and S. Goldwater. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *North American Chapter of the Association for Computational Linguistics Human Language Technologies (NAACL-HLT)*, pages 139–146, 2007.

[13] F. Lerdahl and R. Jackendoff. *A Generative Theory of Tonal Music*. MIT press, 1985.

[14] W. Maler. *Beitrag zur Durmolltonalen Harmonielehre I (7th ed.)*. F. E. C. Leuckart, 2007.

[15] E. Nakamura, M. Hamanaka, K. Hirata, and K. Yoshii. Tree-structured probabilistic model of monophonic written music based on the generative theory of tonal music. In *IEEE ICASSP*, pages 276–280, 2016.

[16] J. F. Paiement, D. Eck, and S. Bengio. A probabilistic model for chord progressions. In *ISMIR*, pages 312–319, 2005.

[17] J. F. Paiement, D. Eck, and S. Bengio. Probabilistic melodic harmonization. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 218–229, 2006.

[18] G. Papadopoulos and G. Wiggins. AI methods for algorithmic composition: A survey, a critical view and future prospects. In *AISB Symposium on Musical Creativity*, pages 110–117, 1999.

[19] R. D. Prisco and R. Zaccagnino. An evolutionary music composer algorithm for bass harmonization. *Applications of Evolutionary Computing*, pages 567–572, 2009.

[20] D. Quick. Learning production probabilities for musical grammars. *Journal of New Music Research*, 45(4):295–313, 2016.

[21] S. A. Raczyński, S. Fukayama, and E. Vincent. Melody harmonization with interpolated probabilistic models. *Journal of New Music Research*, 42(3):223–235, 2013.

[22] H. Riemann. *Harmony Simplified: or, The Theory of the Tonal Functions of Chords*. Augener, 1896.

[23] M. Rohrmeier. Mathematical and computational approaches to music theory, analysis, composition and performance. *Journal of Mathematics and Music*, 5(1):35–53, 2011.

[24] R. Scholz, E. Vincent, and F. Bimbot. Robust modeling of musical chord sequences using probablistic N-grams. In *IEEE ICASSP*, pages 53–56, 2009.

[25] I. Simon, D. Morris, and S. Basu. Mysong: automatic accompaniment generation for vocal melodies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 725–734. ACM, 2008.

[26] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. D. Roure, and J. S. Downie. Design and creation of a large-scale database of structural annotations. In *ISMIR*, pages 555–560, 2011.

[27] M. J. Steedman. A generative grammar for jazz chord sequence. *Music Perception*, 2(1):52–77, 1984.

[28] M. Towsey, A. Brown, S. Wright, and J. Diederich. Towards melodic extension using genetic algorithms. *Educational Technology & Society*, 4(2):54–65, 2001.

[29] K. Yoshii and M. Goto. A vocabulary-free infinity-gram model for nonparametric bayesian chord progression analysis. In *ISMIR*, pages 645–650, 2011.