Vol.2015-MUS-107 No.25
2015/5/23

楽曲パターンマイニングに基づく楽曲編集ツールの試作

浅田 学史[1,a)]　清水 豪[†1,b)]　竹内 和広[†2,c)]

概要：コンピュータを用いて音楽作成や編集を行う作業は DTM(Desk Top Music) と呼ばれる．音楽の世界では，長い間，楽譜を用いて作曲，編曲，演奏などがなされてきたため，それらの作業がコンピュータ化された現在であっても，多くの DTM ソフトウェアは，長年蓄積されてきた楽譜データの概念を前提に作成されている．そのため，DTM ソフトウェアは，楽曲の編集の単位は，最も基本的な記号（音符）か小節となっている．本研究では，DTM ソフトウェアでは，紙の楽譜の上では不可能であった，楽曲の小部分に対するコピーアンドペースト，音を聞きながらの編集，演奏のアーティキュレーションの微調整などが行えるため，小節以外の複数の記号をまとめて扱える単位を導入した楽曲編集ツールの試作を行う．

1. はじめに

近年，歌声合成ソフトウェアが成功し，動画投稿サイトや音楽市場で作品が流通するようになっている．また，多くの楽曲において，人が楽器を演奏する生演奏だけではなく，コンピュータを使った自動演奏も標準的に用いられるようになっている．さらに，人が演奏する楽器や歌唱であっても，その記録はアナログ信号で行われるのではなく，ディジタル情報として保存，編集がなされるようになってきた．このような背景から，音楽に対する情報処理の研究が盛んになされている．例えば，ユーザの意図を何らかの形で反映できる作曲支援 [1][2] や，楽器が演奏できなくても計算機上の操作で思い通りの演奏が可能な演奏支援 [3] などが取り組まれている。などが取り組まれている．しかし，これらの研究の多くは編集単位として音符を採用しており，適切な分割単位については議論していない．さらに，流通しているソフトウェアにおいても，非常に長い間，楽譜を用いて作曲，編曲，演奏などがなされてきたため，長年蓄積されてきた楽譜データの概念を前提に作成されている．例えば，DTM（DeskTop Music）ソフトウェアは，それぞれ独自のインタフェースを採用して構成されているものの，楽曲の編集の単位は，最も基本的な記号（音符）か小節となっている．ここで，楽譜は，特定の音が一定の長さが続くことを示す音符という記号を，基本単位とし，音符がどの順に，どのような方法で演奏されるかを記述した記号の系列データである．また，1 人の歌唱，あるいは，1 つの楽器で演奏される楽曲ではなく，合唱や合奏形式の楽曲であれば，ボーカルと，伴奏のギターやピアノ，ドラムといったパート（それぞれ，ボーカルのパート，ギターのパートなどと呼ぶ）の音符系列を時間軸上に同期する複数の音符系列として表現される．なお，小節は楽曲を時間的長さで等分した時間的単位で，最小記号である音符の連続的まとまりで扱うのに用いる一般的な単位である．他方，DTM ソフトウェアでは，紙の楽譜の上では不可能であった，楽曲の小部分に対するコピーアンドペースト，演奏のアーティキュレーションの微調整などが行えるため，小節以外の複数の記号をまとめて扱える単位が必要である．メロディの繰り返し，変形に着目して，変化のパターン化を試みた研究 [4] では，音程の変形パターンを分類し，フレーズを繰り返してから変形しての解析を試み，変形とされるものの大部分が単純なシフトを伴った変形であることを示している．この研究では，人手によるフレーズのアノテーションされているメロディのコーパスが用いられ，ボーカルパートでかつ，音程の変化のみで分析を行っている．本稿では，音符を 1 つずつ基本編集したり，伝統的な単位である小節を単位として基本編集したりするよりも，基本単位である音符を複数個まとめた系列単位で追加，削除，複製することによる有用性について検討する．具体的な手法として，本稿では，楽譜を文字系列データとして表現する方法を示した上で，データ構造と音符系列からの系列単位

[1] ジャトー株式会社
JATO Co.,Ltd
[†1] 現在，大阪電気通信大学大学院工学研究科
Presently with Osaka Electro-Communication University, Graduate School of Engineering
[†2] 現在，大阪電気通信大学情報通信工学科
Presently with Osaka Electro-Communication University, Department of Information and Communication Engineering
[a)] s-asada@jato.co.jp
[b)] mi15a002@oecu.jp
[c)] takeuchi@isc.osakac.ac.jp

ⓒ 2015 Information Processing Society of Japan

を抽出するために，n-gram や PrefixSpan 法などを用いて系列の記述を行い，系列マイニングを行う指標として，圧縮率・パート推定の判別率という指標で検討を行う．圧縮率は符号化の考えを用いており，パターンの辞書により記号列を 1 つの符号に置き換え，効率的に符号長を短くできるかを評価する指標である．

さらに，決定木を用いて，ボーカルパートにおけるパターンが出現する規則を調査した．発見した規則により，楽譜を提案する単位を検出し，編集を可能にする楽曲編集ツールを試作する．ここで，楽曲の編集とは，作曲や編曲，自動演奏を行うために，音符系列を作成することである．音符系列は，1 から作成しなくても，既存の音符系列に音符を追加，削除，複製（この 3 つを基本編集と呼ぶ）することによっても実現できる．

2. 楽曲の表現方法

2.1 楽曲の文字列化

コンピュータで操作するために楽曲の多くは，MIDI という規格で保存されている．MIDI 規格はどの音をどの時刻にどれだけの長さを発生させるかを記述するための規格であり，リアルタイム性・同期性を重視している．また，MIDI 規格では区切りなどは一切なく，音符の系列である．このような楽曲データを系列として扱うために，図 1 のように，「音高」「音価」「音高・音価」「音名」「音名・音価」という文字列として音符を表現する．数値がパラメータを，文字が種類を表している．音価は，1 を 16 分音符として，その倍数で表している．

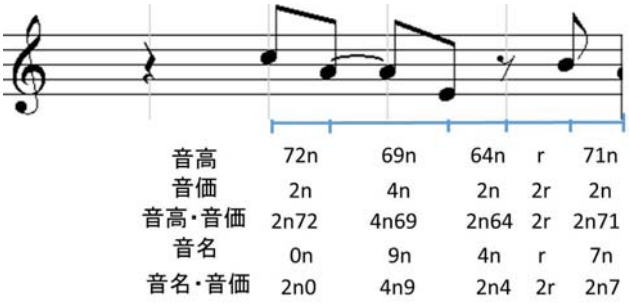

図 1 音符を文字列化した例

2.2 和音の扱い

パートによっては和音が存在する．和音は 2 つ以上の音符が重なったものである．本研究では，パートごとに異なる和音を文字として扱うために，7 種類の楽器の 6 万パートから和音辞書を作成した．和音辞書を用いることで，一定の重なりの和音を 1 つの文字としてパターンに組み込むことができる．

これにより，和音と単音の混合であっても単音と同様に扱うことができる．音価の例を図 2 に示す．ここで“C”が和音を示しており，先頭の数字が和音数，最後の数字が辞書により添付された ID を示している．

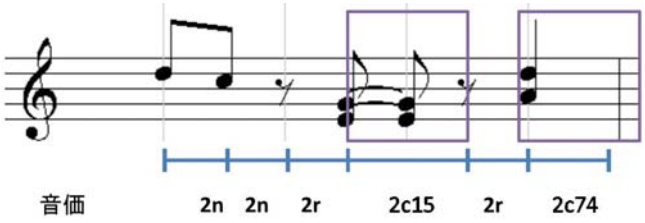

図 2 和音の文字列化の例

2.3 楽曲データ

本研究では 50,000 Midi Files For All Genres Of Music をデータとして用いる．これらのうち，キーがないものや，4/4 拍子ではないもの，テンポがないものを取り除いた 38,432 件を用いる．

3. 楽曲パターンマイニング

文字列化した楽曲データに対してパターンマイニングを行う．ここで，パターンマイニングによって得られたパターン集合を $P = m_{i,1}, \cdots, m_{i,k}, \cdots, m_{i,\#P}$ とする．また，用途に適したパターンを検討するため，複数の方法を用いた．楽曲に対して，「小節線」「 n-gram 」「PrefixSpan(頻度)」「PrefixSpan(信頼度)」を用いて楽曲パターンマイニングを行う．よって，文字列表現 (5 通り) × マイニング手法 (6 通り) の 30 通りを検討する．

3.1 小節線

小節線区切りとは，図 3 のように，小節線により等間隔に区切られた区間をパターンとして取り出したものである．

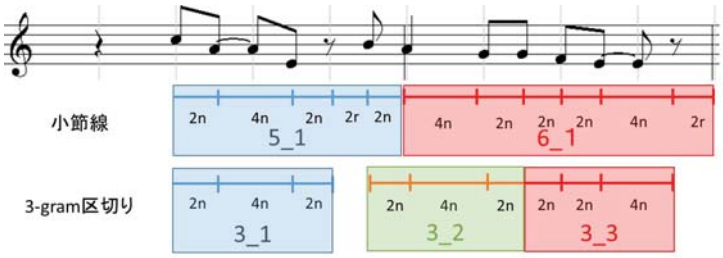

図 3 小節と 3-gram によるパターン化の例

3.2 音符 n-gram

n-gram とは音符列を n という定数間隔で区切り，パターン化したものである．n=3 の例を図 3 に示す．事前実験において，5 以上のパターンの頻度が著しく低かったため，n は 2-4 に設定する．

ⓒ 2015 Information Processing Society of Japan

3.3 PrefixSpan

PrefixSpan[5] は連続した系列データからパターンを取り出す系列パターンマイニングの一種である．系列を構成する最小要素集合を $\Sigma = \{e_1, e_2, \cdots, e_{\#\Sigma}\}$ と表し，e_i をアイテムと呼ぶ．このとき，時系列のデータ列 S をシーケンスと呼び，$S = \{s_1, s_2, \cdots, s_{\#S} \in \Sigma\}$ と表す．また，シーケンスのデータベースをトランザクションと呼び，$T = \{S_1, S_2, \cdots, S_{\#T}\}$ と表す．さらに，出現するパターンの集合を $P = \{p_1, p_2, \cdots, p_{\#P}\}$ と表す．PrefixSpan は射影データベースという考えを取り入れ，メモリを効率的に使用するアルゴリズムで，木構造で表現されたパターンを得ることができる．このアルゴリズムは最大の深さを系列の長さとし，探索空間を限定している．イメージ図を図 4 に示す．

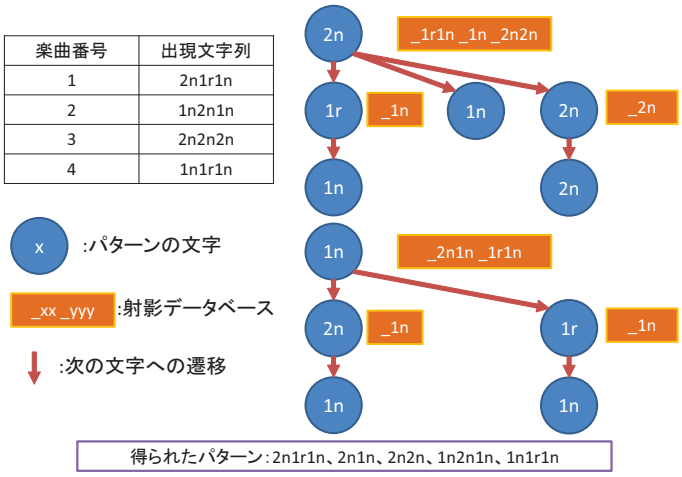

図 4　PrefixSpan のイメージ図

系列パターンマイニングでは固有かどうかの判定には，式 1 で定義される頻度（supp）が利用されており，頻出する系列パターンを特徴的な系列パターンとして発見する．従来の系列データマイニングでは，比較的短い複数のデータ系列から構成されるデータベースを対象として，部分系列 p が何個の系列 S の中に出現するかを問題としてきた．すなわち，与えられた部分系列の頻度が，指定した頻度以上となる部分系列を，特徴的な系列パターンとして発見している．

$$\mathrm{supp}(S) = \frac{S\text{ を含む系列データ数}}{\text{系列データ数}} \tag{1}$$

ただし，S は部分系列を表すとする．

また，発見された系列パターンに対して，その部分系列パターン問の信頼度を評価することにより，系列パターンのルール化も行っている．このとき，系列パターンにおける信頼度（conf）は式 2 に示すように定義されている．本式においては，S_p が系列パターン S に含まれる部分系列パターンを表している．

$$\mathrm{conf}(S|S_p) = \frac{S\text{ を含む系列データ数}}{S_p\text{を含む系列データ数}} \tag{2}$$

PrefixSpan のアルゴリズムを以下に示す．

(1) 系列集合 T のすべての長さが 1 のアイテムの集合を c とし，$k = 1$ とする
(2) 集合 c の中から最小頻度を満たすアイテム順番に 1 つ取り出し，l_1 とする
(3) 系列パターン l_k を用いて射影データベース SDB を作成する
(4) 集合 c のアイテムで拡張可能なパターンの頻度を射影データベース DB より調べ最小頻度を満たすアイテムを l_{k+1} とする
(5) 射影データベース DB を更新し $k = k + 1$ とし，系列 l_{k+1} が作成できなくなるまで”ステップ 3 にもどる

ここで，制約として最大パターン長を 7 とし，最大隣接距離を 1 とすることで連続した長さ 7 までのパターンを抽出している．PrefixSpan ではパターンを Prefix Tree という木構造を得ることができる．この木にはそれぞれのノードに頻度の情報が添付されており，複数の評価方法を用いて，木生成後に足切りをして複数のパターンを得ることができる．そのため，PrefixSpan の処理時はあえて最少頻度などの厳しい制限は加えていない．本研究では，パターンのランキングの際に頻度を用いた“PrefixSpan（頻度）”と，信頼度を用いた“PrefixSpan（信頼度）”の 2 種類を用いる．

4. 楽曲に対するパターンの適用

Viterbi Algorithm により，取得したパターンを，実際の楽曲に適用する．Viterbi algorithm とはテーブルを作成し，ある評価基準が最大となるルートを選択する方法である．本研究ではパターンの適応により，最も長さが短くなる，すなわち圧縮率が最も高くなるように適応する．また，頻度や信頼度によりランキングを行い，上位 100 パターンを対象のパターンとする．

5. 複数尺度によるパターン化の評価

5.1 圧縮率

提案した複数のパターン化手法の有効性を比較するため，圧縮率の平均値により比較する．ここで，圧縮率とは以下で定義する．

$$\text{圧縮率} = \frac{\text{適応されたパターン数+パターンではなかった音符数}}{\text{音符総数}} \tag{3}$$

ボーカルパートとドラムパートの圧縮率の平均値を表 1，表 2 に示す．表 1 より，ボーカルパートは音程・音価と音名・音価で，ほとんど圧縮できていない．また，固定長の手法に対して，可変長である PrefixSpan の圧縮率が高かった．表 2 より，ドラムパートは他のパートと比べ小節線での圧縮の効率が高かった．

ⓒ 2015 Information Processing Society of Japan

表 1　ボーカルパートの圧縮率の平均値

		圧縮率				
		音高	音価	音高・音価	音名	音名・音価
マイニング手法	小節線	0.973	0.913	0.989	0.946	0.987
	2-gram	0.793	0.707	0.916	0.572	0.888
	3-gram	0.865	0.601	0.963	0.807	0.928
	4-gram	0.889	0.709	0.982	0.847	0.962
	PrefixSpan(頻度)	0.710	0.567	0.947	0.555	0.905
	PrefixSpan(信頼度)	0.731	0.584	0.960	0.585	0.934

表 2　ドラムパートの圧縮率の平均値

		圧縮率				
		音高	音価	音高・音価	音名	音名・音価
マイニング手法	小節線	0.683	0.744	0.851	0.652	0.839
	2-gram	0.659	0.914	0.966	0.637	0.960
	3-gram	0.611	0.593	0.827	0.584	0.792
	4-gram	0.617	0.885	0.974	0.596	0.969
	PrefixSpan(頻度)	0.567	0.569	0.857	0.508	0.839
	PrefixSpan(信頼度)	0.574	0.586	0.864	0.518	0.847

5.2　パート判別率

ドラムやボーカルなどのパートによる，適切なパターン集合の差異を調査するために，パート推定の判別率を比較する．各パートごとに獲得したパターン集合の有無を特徴量とし判別率を比較する．この判別率により，パートごとに抽出したパターン集合がそのパート特有であり，このような，パートごとの有効なパターンを調査する．

例として，ボーカルパート推定を例に実験方法を説明する．6930 件のボーカルパートデータに“vocal”というラベルを付与し，ボーカルパート以外の 3317 件に“other”というラベルを付与する．10217 件の楽曲 $M = m_1, \cdots, m_i, \cdots, m_{10217}$ から，ある楽曲 m_i がボーカルパートか否かを推定する SVM モデルを構築する．モデル構築における特徴ベクトルは，各楽曲 m_i に対して各パターン $p_{i,k}$ の有無を表現した特徴ベクトル $V = v_1, \cdots, v_i, \cdots, v_{100}$ を bag of words 法（式 4）に基づいて作成する．bag of words の特徴として，出現の有無の情報のみを特徴ベクトル化しているため，出現の順序は考慮しない．これらの条件で十交差検定を行う．また，本研究では，SVM の実装には R に提供されている「e1071」パッケージを用いる．

$$v_{i,k} = \begin{cases} 1 & (p_k \subseteq m_i) \\ 0 & (otherwise) \end{cases} \tag{4}$$

同様の実験を各パートで行う．ボーカルパートとドラムパートの判別率の平均値を表 3，表 4 に示す．表 3 より，ボーカルパートは固定長のパターンより可変長の PrefixSpan の判別率が高かった．表 4 より，ドラムパートは小節線での判別率が他パートと比べ，判別率が高かった．これは小節線上の特徴的な配置をパターンで表現できているといえる．また，音程による判別率が高かった．これは，スネアやハイハットなどの固定音程で表現するドラムのパターンが特徴的であったからであると思われる．

表 3　ボーカルパートの判別率の平均値

		判別率				
		音高	音価	音高・音価	音名	音名・音価
マイニング手法	小節線	0.676	0.705	0.676	0.714	0.688
	2-gram	0.835	0.708	0.769	0.741	0.676
	3-gram	0.759	0.684	0.676	0.727	0.676
	4-gram	0.713	0.676	0.676	0.724	0.675
	PrefixSpan(頻度)	0.842	0.718	0.730	0.750	0.682
	PrefixSpan(信頼度)	0.834	0.707	0.678	0.733	0.686

表 4　ドラムパートの判別率の平均値

		判別率				
		音高	音価	音高・音価	音名	音名・音価
マイニング手法	小節線	0.759	0.724	0.668	0.718	0.669
	2-gram	0.816	0.778	0.656	0.828	0.653
	3-gram	0.683	0.803	0.815	0.728	0.778
	4-gram	0.607	0.685	0.590	0.716	0.579
	PrefixSpan(頻度)	0.831	0.816	0.718	0.750	0.707
	PrefixSpan(信頼度)	0.839	0.817	0.707	0.733	0.711

6.　パターン出現規則の調査

パターンを発見し辞書を作成することができたが，未知の楽曲に対してのパターン表現は辞書内に存在するパターンに限定され，楽曲において稀に出現するパターンは表現することができない．そこで，パターンの前後には一定の規則が存在するという仮説の基で，未知のパターン候補に対してパターンか否かを判断する条件式を 6930 件の楽曲から決定木のアルゴリズムである CART を用いて記述する．決定木の実装には python ライブラリ scikit-learn を用いた．条件式記述における特徴ベクトルは，直前の音符または休符の音価，音符の音高差を用い，文字列での表現方法は音高・音価，マイニング手法は PrefixSpan（頻度）を用いた．また，パターン数は 1000 とした．

決定木でのパターン開始時の条件式記述結果の一部を図 5 に，決定木でのパターン終了時の条件式記述結果の一部を図 6 に示す．最初の条件はパターン開始時，パターン終了時共に音符の音価が条件となっており，パターン開始時は直前の音符の音価が付点 8 分音符以上，パターン終了時には直前の音符の音価が付点 16 分音符以上という条件式で記述されていた．パターン開始時とパターン終了時での条件式の比較を行った結果，パターン開始時の条件式にはパターン終了時に比べ音符や休符の音価での条件式が頻出しており，パターン終了時にはパターン開始時に比べ音符の音高差での条件式が頻出していた．また，パターン終了時においては，音符の音高差は 9.5，6.5，11.5 など楽曲において大きいと考えられる音高差が見られた．このことから比較的長い音符や休符の音価が出現したタイミングでパターンが開始され，音符の音高差が大きく開くタイミングでパターンが終了するのが楽曲における一般的なパターンだと考えられる．

ⓒ 2015 Information Processing Society of Japan

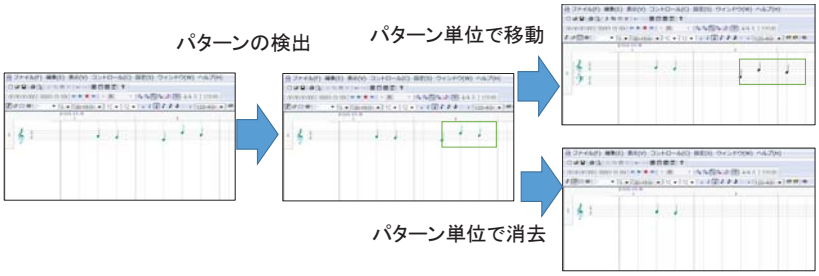

図 7　楽曲編集ツールの利用例

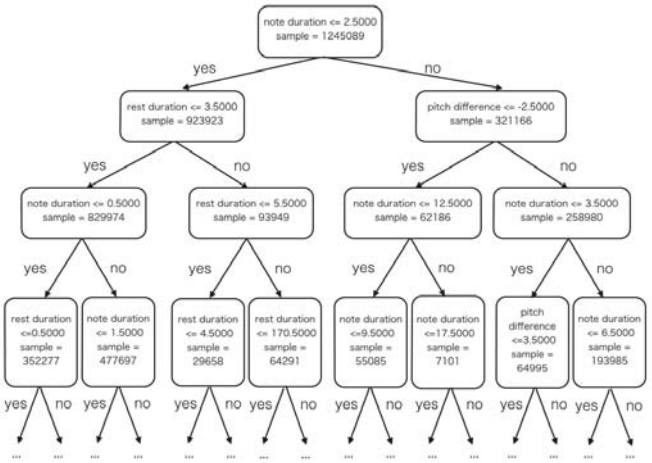

図 5　決定木でのパターン開始時の条件式記述結果の一部

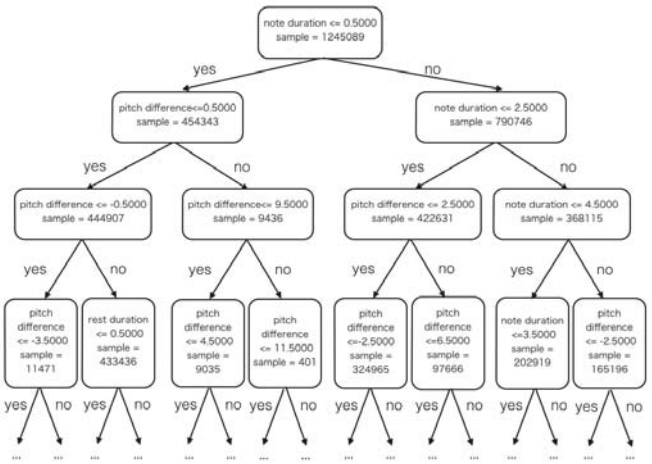

図 6　決定木でのパターン終了時の条件式記述結果の一部

7. 楽曲編集ツールの試作

獲得したパターンが出現規則を用いて楽譜のパターンの検出・編集が可能な楽曲編集ツールを試作する．本ツールはオープンソースの MIDI シーケンサーソフト『世界樹』[*1] をベースにパターンの検出と追加，削除，複製などの編集を実装した．本研究では実験的にボーカルパートのパターンの検出・編集を実装した．楽曲編集ツールの利用例を図 7 に示す．

[*1] http://openmidiproject.sourceforge.jp/Sekaiju.html

8. おわりに

音高・音価・音名を用いた文字列化と，小節線・n-gram・PrefixSpan によるパターンマイニング手法を楽曲に適用し，楽曲の編集に適した単位について検討した．圧縮率で適応率を，パート推定により，パートごとのパターンの差異について調査した．メロディを担当するボーカルには可変長のパターンが，リズムを担当するドラムには音程と小節が有効であるという結果が得られた．これは大まかな役割に応じて，パートごとのパターンの変化を表現できたといえる．また，これらのパターンが発生する条件を決定木により調査し，その結果に基づく編集ツールを試作した．

今後の課題として，和音において辞書を用いて ID で管理すると文字の種類が 34 万種類になり，マイニングがうまくいかないことがあるため，より良い表現方法が必要であると考えられる．また，本研究では MIDI データを利用したため，ジャンルや A メロ・B メロなどのアノテーションがない．そのため，音楽としての平均的なパターンは獲得できたが，より厳密なアノテーションが施されたデータによるパターンマニングで，さらに実用的なパターンを得ることができるかもしれない．また，ボーカルパート以外のパートを対応させることも必要である．

参考文献

[1] 嵯峨山茂樹 酒向慎司 堀玄 深山覚：確率的手法による歌唱曲の自動作曲 ([特集] 音楽制作と情報処理の友好関係，システム制御情報学会 (2012).
[2] Fukayama S. and Saito D. and Sagayama S.: ASSISTANCE FOR NOVICE USERS ON CREATING SONGS FROM JAPANESE LYRICS, Proc.of ICMC 2012 (2012).
[3] 馬場隆 橋田光代 片寄 晴弘：実際にオーケストラを指揮する感覚に焦点を当てた指揮システム VirtualPhilharmony，情報処理学会研究報告. [音楽情報科学](2010).
[4] 船越咲 梅村祥之: 隣接フレーズ間のメロディの変化パターンの分析，情報処理学会研究報告. [音楽情報科学](2014).
[5] Pei Jian and Han Jiawei and Mortazavi-Asl Behzad and Pinto Helen and Chen Qiming and Dayal Umeshwar and Hsu Meichun: PrefixSpan: Mining Sequential Patterns by Prefix-Projected Growth，IEEE Computer Society(2001).

ⓒ 2015 Information Processing Society of Japan