音符長に注目した擬音による楽曲検索手法の提案

石原 健司† 木村 文則†† 前田 亮††

†立命館大学情報理工学研究科 〒525-8577 滋賀県草津市野路東 1-1-1

††立命館大学情報理工学部 〒525-8577 滋賀県草津市野路東 1-1-1

E-mail: †is003085@ed.ritsumei.ac.jp, ††fkimura@is.ritsumei.ac.jp, ††amaeda@media.ritsumei.ac.jp

あらまし 現在の楽曲検索システムには，アーティスト名や歌詞といった楽曲の情報を利用するものが多い．しかしユーザがこれらの情報を持っていない場合，曲を聴いただけではクラシック音楽や BGM などの歌詞を用いない楽曲に対する検索は困難である．本論文では擬音をクエリとした楽曲検索手法を提案する．本手法では，あらかじめ検索対象とする楽曲内の音符長の変化について索引を作成する．また楽曲と同様に入力クエリから音符長の変化を取得する．クエリの先頭の音符長変化が一致する楽曲対象部に対して DP マッチングを行い，その類似度を基にランキングを行う．本論文では提案手法の評価実験を行い，考察を行う．

キーワード 楽曲検索，擬音，DP マッチング

A Method for Music Retrieval by Onomatopoeia Based on Note Length

Kenji ISHIHARA† Fuminori KIMURA†† and Akira MAEDA††

†Graduate School of Information Science and Engineering, Ritsumeikan University

1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577 Japan

††College of Information Science and Engineering, Ritsumeikan University

1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577 Japan

E-mail: †is003085@ed.ritsumei.ac.jp, ††fkimura@is.ritsumei.ac.jp, ††amaeda@media.ritsumei.ac.jp

Abstract When we use a music retrieval system, we need information of music such as singers or music lyrics. However, if we do not know these music information, it is difficult to search music, especially for instrumental music. In this paper we propose a method for retrieving music by onomatopoeia. With the proposed method, we create an index on change of note length in all music of the search target in advance. Also, indexing on change of note length in query. This system calculates similarity of each music and query by DP Matching, only part of the music that matches change of the note length at the top of the query. Last this system ranks music based on similarity. This paper shows the result of evaluation by the proposed method.

Keyword Music Retrieval, Onomatopoeia, DP matching

1. はじめに

現在の楽曲検索システムには歌詞やアーティスト名などのメタデータを利用して曲を絞り込むキーワード検索や，マイクを通して自らが歌うことで目的の曲を検索する音声検索がある．しかしメタデータで楽曲を絞り込むキーワード検索では，ユーザが歌手や歌詞などの楽曲に関する情報がわからない場合や，歌詞が用いられないインストゥルメンタル曲を検索することは困難である．また音声検索では，音程情報を利用した高精度の検索が可能であるが，公共の場で歌うことが憚られる場合やユーザが音程を正しく歌えない場合には目的の曲を検索することは困難である．

これらの問題解決のために，我々は楽曲の特徴を感覚的に表現することが可能な擬音に着目し，擬音の文字列でメロディを入力することで曲を検索する手法を提案した[1]．以下この手法を「擬音手法」と記述する．

擬音とは一般的にパチパチやシトシトなどの物が発する音やその状態を字句で表現したものである．他人に曲を伝える際に，曲のフレーズを擬音により表現することが多いため，本論文ではユーザが曲をハミングした際，その音を「ラーラーラーーーララー」のように字句で表現したものを擬音と定義する．擬音で表現可能な楽曲情報はリズムのため，正しく音程を表現できないユーザの利用が考えられる．
検索対象にはインストゥルメンタル曲で，入手が容易であるクラシック音楽の MIDI データ[2]を利用する．

2. 関連研究

キーワード検索や音声検索以外にも様々な楽曲検索手法が提案されている．梶ら[3]による手法では，楽曲に対する人間の解釈をアノテーションとして収集し，それらを利用することで楽曲を絞り込んでいる．人間の感性を利用することで，楽曲に対する情報が曖昧な場合にも検索が可能である．

斧山ら[4]の手法では，楽曲のフレーズを入力に用いることで類似楽曲の検索を実現している．データベース内の楽曲から作成したフレーズ内の音楽特徴を利用することでユーザのイメージする楽曲を絞り込んでいる．

池谷ら[5]の手法では，打鍵により入力したリズムを用いて楽曲を検索する．印象的なメロディを打鍵で入力することで得られるリズムの特徴を利用している．リズムを用いる点では我々の擬音手法と類似しているが，池谷らの手法では楽曲の特徴的なフレーズを手動で選択し収集する点が異なる．

3. 擬音での楽曲検索手法

3.1 概要

今回の提案手法では，擬音をクエリとする点は我々が以前に提案した手法[1]と同様であるが，音符長の変化に注目したマッチングを行うことで検索精度と処理速度の高速化を図る．以下この手法を「音符長変化手法」と記述する．あらかじめ Web から収集した MIDI データに対して，音符長の変化毎に索引を作成する．ユーザのクエリに対しても同様の処理を行い，音符長の変化を利用したマッチングを行う．マッチングにより算出した各楽曲との類似度を基にランキングを行い，検索結果としてユーザに提示する．

3.2 従来手法の問題点

従来手法である擬音手法では，ユーザの入力する擬音文字列（例：ラーーラーラーーーラ）とのマッチングを行うために，MIDI データを擬音文字列へと変換していた．擬音への変換では，ユーザへのアンケートを基に，2 分音符なら「ラーーー」，8 分音符なら「ラ」のように音符の種類毎に擬音を決定していた．しかし擬音文字列同士でのマッチングには様々な問題がある．

第一の問題は，マッチングの結果がユーザの感性に強く依存することである．MIDI の変換規則として定めた音符と擬音の対応にユーザの感性が一致していれば精度は良くなるが，違う場合には精度は悪くなる．

第二の問題は，テンポの変化への対応が困難な点である．テンポが異なる楽曲において，同じ 4 分音符でもユーザの感じ方は変化する．テンポの変化について，4 分音符を「ラ」と「ラー」のどちらに変換するか，境界線を定めることは困難である．

第三の問題は，計算量の問題である．擬音手法では一音符ずつずらしながらマッチングを行い，網羅的にユーザの検索対象部を探索するため，処理にかかる時間が長いことが問題であった．MIDI データ自体の長さやクエリの長さにより変動するが，検索対象の楽曲数が 50 の場合，結果の提示までに 7 分[1] 程度の時間が必要であった．多くの楽曲を扱うことが想定される楽曲検索システムにおいてこの処理時間は大きな問題である．

3.3 全体の流れ

図 1 に音符長変化手法の流れを示す．前処理として Web から収集した MIDI データに対して音符長の変化毎に中間言語に置き換え，索引を作成する．ユーザの問合せ時に，クエリである擬音文字列を MIDI データの変換と同様に中間言語に置き換える．クエリの先頭の音符長の変化を確認し，楽曲側で同様の音符長変化が起きる箇所にのみマッチングを行い類似度を計算する．各楽曲に対して類似度の計算を繰り返し，算出した類似度を用いてランキングを行った後，ユーザに結果を提示する．MIDI データの変換，マッチング処理についてはそれぞれ 4 章，6 章で述べる．

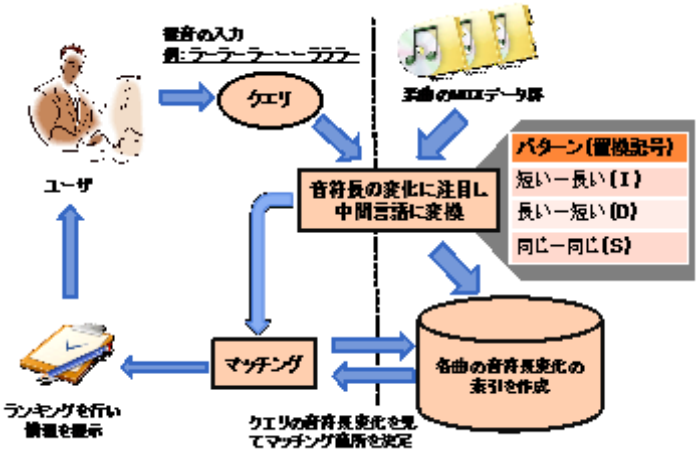


図 1 システムの流れ

[1] CPU：Intel Core i5 (3.30GHz)，メモリ 4GB の PC の場合
開発には Java 言語を使用

4. MIDI データへの処理

4.1 MIDI データのテキスト形式への変換

MIDI データには「どの位置でどの音がどれだけの強さで鳴るか」といった情報が特定のフォーマットで記述されており，WAV などの波形データよりも情報の抽出が容易である．これらの情報を扱うために，MIDI データ内の音符の変化を時系列順にテキスト形式に変換する．ある MIDI データの例を図 2 に示す．図 2 は Domino[2] という MIDI 再生ソフトウェアで MIDI データを見やすい形式で表示したものである．縦軸が音程を表しており，横軸が時間を表している．4 分音符なら「480」，8 分音符なら「240」のように音の長さで値が設定されている．

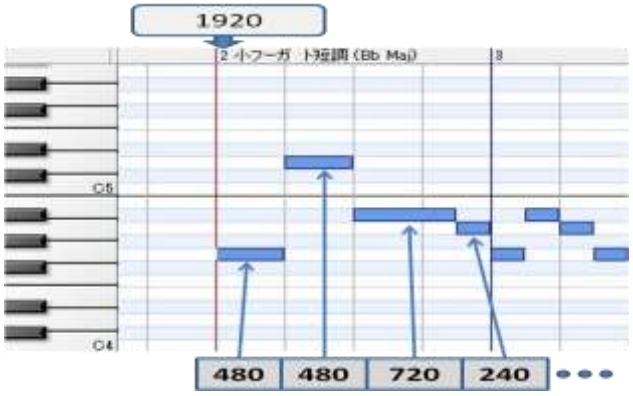


図 2 MIDI データの例

4.2 人の視聴と実際の音符長の違い

単純に MIDI データ内の音符長を利用した場合，ユーザの感じる音符長とは異なる場合がある．図 3 の左に示した MIDI イベントは，「2 分音符，付点 4 分音符，4 分音符」の連続であるが，実際にユーザが視聴し，擬音で表現する場合，図 3 の右のようになると考えられる．付点 4 分音符の途中で 4 分音符が鳴り始めるため，次の音符が鳴り始めるまで鳴っていた音符長が擬音入力の際に利用される．

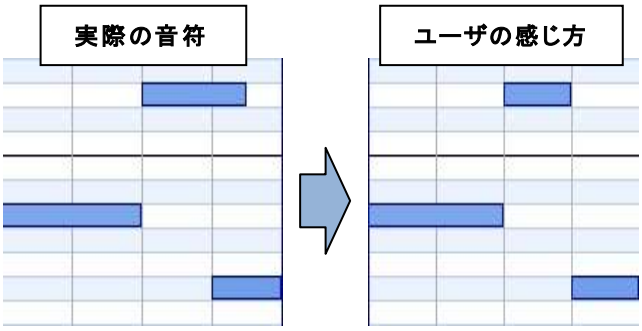
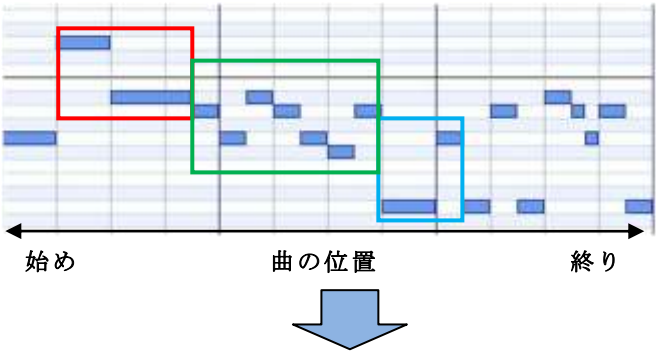


図 3 MIDI イベントの重なり

[2] http://takabosoft.com/domino

4.3 索引の作成

本手法で利用する楽曲の情報は，音符の鳴り始めた位置と音符長である．楽曲がどのような音符の連続で構成されているかという点に注目し，音符長の変化を見ることで簡略化を行う．隣接する 2 つの音符を比較し，1 つ目が短く 2 つ目が長い場合には「I」，逆の場合は「D」，2 つの音符が同じ長さの場合は「S」のように記号に置き換える．音符長の変化を表す 3 つのパターンについて各楽曲に対して索引を作成する．その際，楽曲内には同じ長さの音符が連続することが多いため「S」が連続して現れる部分を「S3」「S10」のようにまとめ，その先頭の出現位置だけを索引に登録する．図 4 に索引の一例を示す．図 4 内の表には，図 4 内の MDI データでそれぞれの音符長の変化が表れる位置を先頭からの音符数で表記している．



パターン（置換記号）	音符長変化の位置
短い－長い（Ｉ）	2,10,18,…
長い－短い（D）	3,11,16,…
同じ－同じ（S）	1,4,12,17,19,...

図 4 音符長の変化の索引

5. クエリへの処理

5.1 アンケート

曲を聴いた際に人がどのように擬音でメロディを表現するか調査するため，日本人 10 人に対してアンケートを行った．アンケートでは 10 曲の楽曲を対象とし，擬音でクエリを作成してもらい，収集したクエリを基に考察を行った．クエリを確認したところ，楽曲を擬音で表現する際，同じ楽曲でも人により音符の長さを表現する長音符（—）の数に変化が見られた．また，音符数の入力ミスも見られ，そのミスはクエリの文字列が長い場合や，同じ長さの音符が連続する場合によく発生していた．このアンケートの結果より，擬音手法で行っていた，擬音文字列同士のマッチング手法は

適していないと考えられる．これは音符長をみて音符を擬音へと変換する際の表現方法が人により異なるためである．

5.2 クエリの変換

3.2 節で述べた擬音手法の問題点と 5.1 節で述べたアンケートの考察より，ユーザの正確な入力が望めない問題への対処として，正確な特徴だけを利用することでクエリを簡略化する．クエリから取得可能な正確な特徴とは音符長の変化である．正確な音符の長さがわからない場合でも，隣接する音符のどちらが長いかを判断することは容易である．そこで 4.3 節で述べたように，隣接する 2 つの音符を比較し記号に置き換える．このように音符長の変化を記号で置き換えることで，音符の具体的な長さを考慮する必要がなくなり，簡略化することが出来る．また同じ長さの音符が連続する場合にはその数を「S」の後に付加する．この数字はマッチングの際の類似度の計算に利用する．

クエリを変換例を図 5 に示す．例（ラーラーラーーーララララ）の場合には「ラ」を 1 つの音符，「ー」を音符の長さとして判断し，先頭は「ラー」と「ラー」で同じため「S1」，次に 1 音符ずらし「ラー」と「ラーーー」で 2 音符目が長いため「I」といった要領で置き換える．

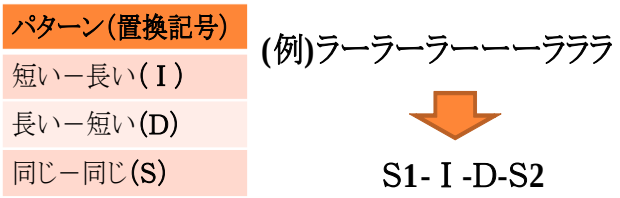


図 5　音符長の変化を表す中間言語

6.　マッチング処理

6.1 マッチングの流れ

4.3 節で述べた，音符長の変化を表す置き換え記号による索引を利用してマッチングを行う．マッチング箇所の決定は，ユーザの入力したクエリの先頭の音符長の変化を見ることで決定する．クエリの先頭が「D」の記号で表される音符長変化である場合，「D」の索引に登録された出現位置からマッチングを行う

本論文では DP マッチングと呼ばれるマッチング手法を改良し類似度の計算に用いる．マッチングを行う楽曲の対象パートは，ドラムなどのリズムパートを除いたメロディを含むトラックに絞る．これはアンケートの結果，ユーザが入力に利用する楽曲対象部は，リズムパートなどの規則的な部分ではなく特徴的なメロディパートが多かったためである．

6.2　DP マッチング

DP マッチング[6]の「DP」とは，動的計画法（Dynamic Programming）の略であり，パターン同士の類似度を計る手法である．音声の分野で利用されることが多いが，本手法では文字列の類似度の計算に利用する．DP マッチングの特徴は，部分的な伸縮を考慮する点であり，文字の一致や補完に対してコストを指定することで計算を行う．

本手法では，文字が一致する場合はコストの増加はなし，文字が不一致の場合には「+3」，文字を補完した場合は「+1」といったようにコストを加算していく．そのため累計コストが小さい程文字列の類似度が高いことを表す．

6.2.1 DP マッチングへのルールの追加

本手法では通常の DP マッチングに加え，独自のルールを加えている．

第 1 のルールとして文字の一致確認の際に「I」と「D」，「I」と「S」の比較では増加するコストを変更している．これは 2 つの音符を比較する場合，ユーザが短い音符，長い音符の連続であると感じた部分が実は長い音符，短い音符の連続である可能性よりも，同じ長さの音符の連続である可能性の方が高いと考えられるためである．そのため「I」「S」の違いではコストの増加が 3 に対して，「I」「D」の違いでは 6 に変更して計算を行う．

また第 2 のルールとして，「S」同士の比較時に新たなコストの増加を設定している．連続する「S」の数の差が小さければ類似しているが，差が大きければ類似していないと考えられる．そのため，「S」の差に 0.3 を乗算し，コストの計算に利用する．「S1」と「S4」の比較であれば，コストは(4-1)×0.3=0.9 となる．ただしユーザの入力は終了した場合でも楽曲内の「S」は連続することが考えられるためクエリの先頭と最後尾では楽曲側の連続する「S」が多い場合にコストを 0 としている．

6.2.2 DP マッチングの流れ

処理の流れを図 6 に示す．図 6 は「S1,I,I,S6,I」と「S1,I,D,S6,I」の 2 つの記号列に対して DP マッチングを行った例である．①であらかじめ文字の不一致部分の増加コストを計算しておき，②で補完のコストも加えた累計コストを導き出す．②の表の左上端の「0」の位置を原点とし，隣接する 3 マスの内のどのマスに進んだ場合最もコストが低くなるか判断を繰り返し右下端まで計算を行う．隣接するマスへの移動では，右と下への移動が文字の補完を表すコスト 1 増加，斜めの移動ではコストの増加はない．図 6 の例では，文字の補完が 2 回，「S6」と「D」による文字不一致が一回行われ，累計コストは 5 となる．

①

	S1	I	D	S6	I
S1	0	3	3	1.5	3
I	3	0	6	3	0
I	3	0	6	3	0
S6	1.5	3	3	0	3
I	3	0	6	3	0

②

	S1	I	D	S6	I
S1	0	4	8	10.5	14.5
I	4	0	7	11	10.5
I	8	1	6	10	
S6	10.5	5	4	5	9
I	14.5	6	11	7	5

累計コスト

図 6　DP マッチングの流れ

6.3　マッチングの手順

マッチングを行う箇所は，クエリの先頭の音符長の変化により決定される．その際マッチングに利用する文字列は，変換後のクエリと同じ長さとする．これはマッチングの対象とする楽曲対象部はユーザが想定する音符数と同じであると考えられるためである．この楽曲から切り出し,マッチングに利用する部分を「音楽片」と呼ぶ.音楽片の幅を固定にすることでユーザの音符数のミスが影響すると考えられるが，ユーザの入力にミスが発生する問題は同じ長さの音符が連続する場合に多いため，連続する「S」の数を一つにまとめ，その数により類似度に違いを出すことで少量のミスに配慮している．

従来手法である擬音手法では楽曲の先頭から音楽片の幅は固定で 1 音符ずつずらしながらマッチングを行うことで網羅的にマッチングを行っていた．正解の音楽片を確実に含んでいることが利点であるが，処理時間が問題であった．そこで「音符長変化手法」ではマッチングを行う音楽片を絞ることで処理速度の向上を図る．

6.4 マッチング回数の削減

6.4.1 マッチング箇所の選択

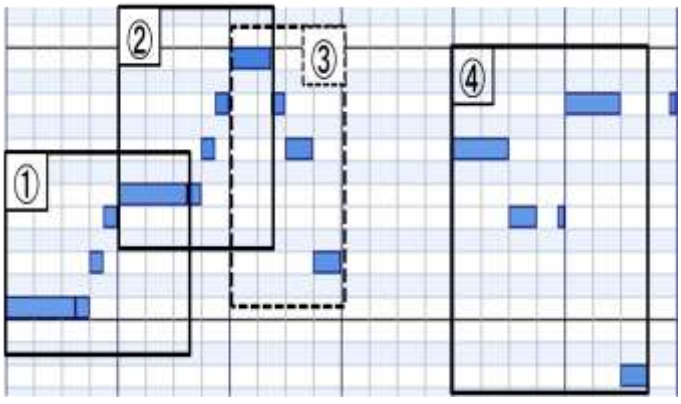
クエリの先頭部分は入力のミスが少ないことを利用し，マッチングを行う音楽片を絞り込んでいる．図 7 では「ラーーラーララーーラー」がクエリである場合，先頭の音符長の変化は「D」であるため，図内の①から④で囲った枠内がマッチングの対象となる．しかし，点線で囲った③の音楽片は，音符の数がクエリよりも少ないため，マッチングは行わない．

6.4.1 同音符長部分の圧縮

楽曲内には同じ長さの音符が連続する部分が多々あり，これを 4.3 節で述べたように 1 つの特徴として表現することで処理回数を減らす方法である．これらの音楽片は擬音で表現しても「ラーラーラーラーラーラーラー」のように単調であるため,ユーザが入力に利用することはないと考えられる．また，入力された場合には完全一致となる楽曲が多数あり，正解の楽曲を提示することは困難である．

6.4.1 楽曲内の休符の考慮

楽曲内にはメロディとメロディの間には度々休符が表れる．この休符でメロディが途切れたとユーザが感じる場合，休符の前後のメロディを同時に入力として利用することは非常に少ないと考えられる．このような休符毎に楽曲を区切ることで，必要な音符数を含まない音楽片へのマッチングを削減することが可能である．メロディの区切れだと感じる休符の長さは人により様々であると考えられるが，今回は全休符以上の長さの休符をメロディの区切れ目とする．その例を図 8 に示す．図 8 の③と④の枠の間には全休符があるためここで楽曲を区切っている．ここで区切らない場合は，③と④の音符が連続しているとみなされるため，③に 2 分音符を 1 つ追加した 5 音符の音楽片に対してマッチングを行う必要がある．



クエリ:ラーーラーララーーラー

図 7　マッチングの手順

7. 評価実験

10 人のユーザにクラシック音楽 10 曲を聴いてもらい，擬音のクエリを作成してもらった．合計 50 件のクエリを用いて，今回提案した「音符長変化手法」と，従来の手法である「擬音手法」で検索を行い，そのランキング結果を基に評価を行う．検索対象として利用する楽曲はクラシック音楽 50 曲である．

7.1 評価指標

ランキングの結果は楽曲内で最も累計コストが低い値を見て決定する．これは累計コストが小さい部分が最もその楽曲内で類似しており，ユーザがクエリに利用した音楽片である可能性が高いからである．また同楽曲内には同じフレーズが繰り返し現れることがあるため，最小累計コストが同じである楽曲は，次に小さい累計コストを確認し，より小さい累計コストを含む楽曲をランキングの上位とする．

手法の評価はランキングの結果をスコアリングすることで行う．ランキング結果の上位 10 件を確認し，正解楽曲が 1 位であればスコアは 1.0，2 位なら 0.9，のように 10 位まで 0.1 ずつスコアを下げ，11 位以下は 0 とする．全てのクエリのスコアを足し合わせ，平均した値を手法のスコアとする．

7.2 実験結果

表 1 にランキングの結果を示す．前述したスコアリングによる両手法のスコアは，音符長変化手法が 0.376，擬音手法が 0.370 となった．50 件のクエリ中，正解楽曲が 3 位までに出現したものは，音符長変化手法では 14 件，擬音手法で 12 件であった．擬音手法よりも音符長変化手法において順位が上がったクエリは 25 件，同順のものが 6 件であった．1 クエリ当たりの平均処理時間[3] は音符長変化手法が 60.2 秒，擬音手法が 415.8 秒となった．

表 1　ランキング結果

	音符長変化手法	擬音手法
スコア	0.376	0.37
正解が1位～3位のクエリ数	14	12
正解が10位より下のクエリ数	23	21
1クエリの平均処理時間(秒)	60.2	415.8

[3] CPU：Intel Core i5 (3.30GHz)，メモリ 4GB の PC を利用

8. 考察

ランキングの結果では，音符長変化手法のスコアが「0.376」，擬音手法が「3.7」となり，スコアの大きな変化は見られなかった．しかし，各クエリでの両手法の結果が変化していなかったわけではない．

8.1 音符長変化手法の利点

擬音手法よりも順位が上がったクエリは 25 件あり，それらのランキング結果を確認したところ，他楽曲との累計コストの違いが大きく表れていた．擬音手法では DP マッチング時のコスト増加条件が少ないため，楽曲毎の最小累計コストを比較する際に同値である楽曲が多い問題が見られた．しかし音符長変化手法では DP マッチングを改良したことで，本来最小コストが同値であった楽曲集合の中でも差が生まれ，より楽曲の類似度合の判断が容易になった．

8.2 音符長変化手法の問題点

音符長変化手法によるランキング順位が擬音手法より下がったクエリは 19 件であった．連続する「S」をまとめることで，その数を類似度の計算に利用しているが，一つの文字にまとめてしまうことで問題が発生しているクエリが存在した．その例を図 9 に示す．図 9 に示した MIDI の音楽片をクエリとして入力したユーザが存在した．クエリは「ラーラララーラララーラララララララララララララーーー」であり，これを中間言語に置き換えると「D,S1,I,D,S1,I,S13,I」となる．しかし MIDI データ側は「S」がさらに連続しており，同音符数でのマッチングを行う場合に「D,S1,I,D,S1,I,S28,I」とのマッチングとなり「S」の数の差が 15 であると判断してしまい累計コストが増大してしまう．そのため楽曲側の「S28」を分割することで，クエリの「S13」部分は完全に一致した部分，クエリの最後の「I」と楽曲側の残りの「S」との文字一致を判定し，マッチングを行う必要があると考えられる．

8.3 両手法で共通の問題点

クエリ自体に含まれる音符数が少ないことが問題である．実際に存在したクエリを例として挙げる．「ラーーーーラーーーーラララーーー」というクエリは音符長変化手法で 14 位という結果であったが，音符長を表す中間言語に置き換えると，「S1,D,S1,I」である．この変化が表れる曲数は多数あり，正解楽曲が完全一致する音楽片を含んでいても，その数が少ない場合ランキングの結果が悪くなってしまう．現在のランキングの決定では最小累計コストが同じの場合その数を，数も同じの場合次に低い累計コストを利用するため，MIDI データ自体の長さを考慮したランキング決定を行う必要があると考えられる．

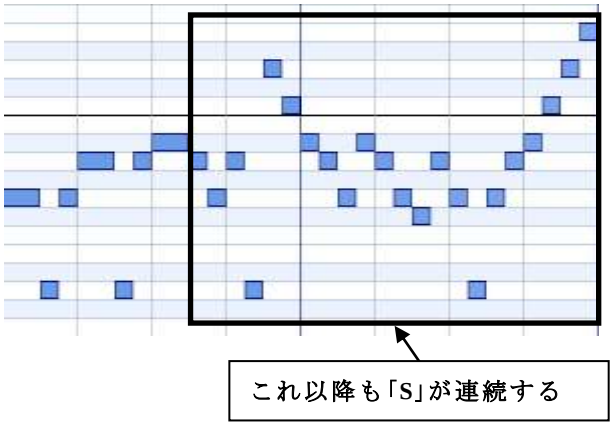


図 2 「S」の変化をまとめる問題点

また別の問題として，クエリとユーザがクエリに利用した楽曲の対象メロディ部を確認したところ，メロディを正確に擬音で表現出来ていないクエリが存在した．これは，前後の音符で長さが違うと感じていても文字での入力に選択肢が少なく結果的に間違った入力となっているクエリである．その例を図 10 に示す．図 10 で 4 分音符を「ラー」と表現した場合，次の 8 分音符は「ラ」と入力される可能性が高い．ここで次に 16 分音符が出てきた場合に，どのように記述するかが問題となる．半角の「ｰ」を利用する可能性が考えられるが，今回クエリの収集を行ったユーザでは，8 分音符も 16 分音符も「ラ」と入力されることが多くみられた．このように「ラー」と「ラ」の中間や，「ラ」より短い状態を表す入力方法がなく「ラ」と記述してしまう問題は，音符長の変化に注目している音符長変化手法では，類似度の低下の原因となる．

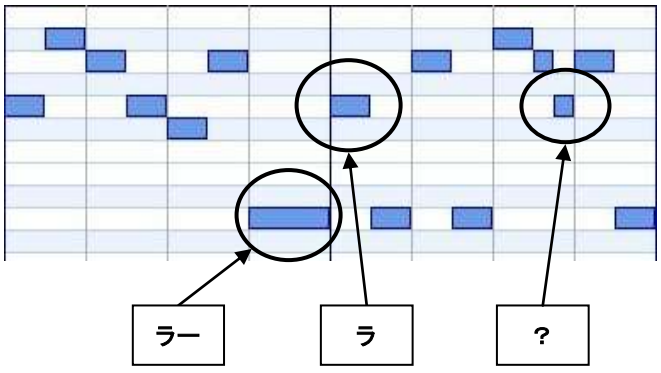


図 3 クエリ入力の問題

8.4 DP マッチングのコスト設定

文字不一致でコストが 3 増加といったパラメータの設定を変更した場合の結果についても検証の必要があると考えられる．

また別に DP マッチング時のコスト増加の設定で少し問題であると感じた点が存在する．それは，現在「S」と「I」または「S」と「D」の比較では「S」に付加された数字が何であっても文字不一致コストとしてコストを「3」増加している点である．この場合「D」と「S」類似を判断する際に「S」が「S1」の場合と「S10」の場合では，増加するコストを変化する必要があると考えられる．

9. おわりに

本論文では擬音文字列をクエリとし，音符長の変化に注目した楽曲検索手法を提案し，評価実験を行った．

本研究の今後の課題は，音符長変化手法における精度の向上である．今回利用した 50 件のクエリの結果を考察することで手法の改善点が判明したため，それらの改善を行い，再実験を行う必要がある．

またクエリのミスを減らすために，ユーザの入力方法に何かしらの制限をかけることも考慮している．クエリから取得できる特徴が少ないため，利用可能な特徴を正確に取得可能な手法やインタフェースの実現が重要であると考えられる．

参 考 文 献

[1] 石原 健司，木村 文則，前田 亮．擬音語入力によるクラシック音楽検索手法の提案．第 4 回データ工学と情報マネジメントに関するフォーラム（DEIM 2012）E3-1，2012．

[2] Classical Music on Classical Archives: http://www.classicalarchives.com/index.html?navID=0

[3] 梶 克彦，長尾 確．楽曲に対する多様な解釈を扱う音楽アノテーションシステム．情報処理学会論文誌 48(1)，-pp. 258-273，2007.

[4] 斧山 青矢，吉田 真一．フレーズ毎の音楽特徴を用いた楽曲検索システムの提案．高知工科大学情報システム工学科 学士学位論文 2011 http://www.kochi-tech.ac.jp/library/ron/2010/2010info/1110381.pdf

[5] 池谷直紀，服部正典，梅木秀雄，大須賀昭彦．リズム入力インタフェース「タタタタップ」による大規模音楽検索．情報処理学会研究報告．27-33，2005.

[6] 内田誠一：DP マッチング概説：基本と様々な拡張，電子情報通信学会技術研究報告(特別講演)，PRMU2006-166，pp. 31-36，2006.