

ATTA: exGTTM に基づく自動タイムスパン木獲得システム

浜中雅俊^{†1} 平田圭二^{†2} 東条敏^{†3}

^{†1}科学技術振興機構 さきがけ研究員

^{†2}NTT コミュニケーション科学基礎研究所

^{†3}北陸先端科学技術大学院大学

m.hamanaka@aist.go.jp

本研究報告では、音楽理論 Generative Theory of Tonal Music (GTTM) に基づき、タイムスパン木を自動で獲得するシステム ATTA (Automatic Time-span Tree Analyzer) について述べる。GTTM は複数のルールから構成されているが、ルールの適用に関して厳密な定義がなされていないため、タイムスパン木を獲得するためには手作業によるルールの適用が必要であった。この問題を解決するため、本研究では、計算機実装用にルールを再形式化した GTTM の計算機モデル exGTTM を提案する。exGTTM の特長は、ルールの優先順位を決めるためのパラメータを導入したことである。exGTTM を実装したシステム ATTA を用いて実験した結果、パラメータの調節により、正解率が向上することが確認できた。

ATTA: Automatic Time-span Tree Analyzer based on exGTTM

Masatoshi Hamanaka^{†1} Keiji Hirata^{†2} Satoshi Tojo^{†3}

^{†1}PRESTO, Japan Science and Technology Agency,

^{†2}NTT Communication Science Laboratories,

^{†3}Japan Advanced Institute of Science and Technology

A.I.S.T. Mbox 0604, 1-1-1Umezono, Tsukuba, Ibaraki 305-8568 Japan

This report describes a music analyzing system called the automatic time-span tree analyzer (ATTA) based on the generative theory of tonal music (GTTM). Previous systems based on GTTM cannot acquire a time-span tree without manual application of most of the rules, because GTTM does not resolve much of the ambiguity that exists with the application of the rules. To solve this problem, we propose a novel computational model of the GTTM called exGTTM that re-formalizes the rules with computer implementation. The main advantage of our approach is that we can introduce adjustable parameters, which enables us to assign priority to the rules. Experimental results showed that after these parameters were tuned, our method outperformed a baseline performance.

1. はじめに

本研究では、音楽理論 Generative Theory of Tonal Music (GTTM) [1] を計算機上に実装する手法について提案する。従来の市販のシーケンスソフトウェアは、音符や休符、コードなど表層的な構造しか操作することができないため、音楽初心者が自分の思い通りに楽曲を操作することは困難である。本研究の最終的な目標は、音楽家の知識を計算機上に実装することにより、音楽という多義的で主観的なメディアを、音楽初心者でも思い通りに扱うことのできるシステムを実現することである。そのための第一歩として、ここでは音楽理論 GTTM を計算機上へ実装する。

音楽理論は、音楽の知識や経験、技能を分析し解釈する方法論を我々に与えてくれるが、その際問題となるのは、分析する上で必要な概念が十分客観的に説明されているかどうかである。これまで様々な音楽理論 [2-4] が提案されてきたが、GTTM は他の音楽理論と比べて比較的厳密なルールで記述されており、音楽知識を形式化する上で最も有望であると我々は考えている。

GTTM の分析の結果得られるタイムスパン木は、曲中の各音の構造的な重要度を階層的に表示したもので、その自動生成は音楽の深層構造の分析 [5][6] を可能とするだけでなく、演奏の表情づけ [7-9] や、音楽情報検索システムの出力結果を提示する際などに有用な、音楽要約 [10][11] を実現

する。これらのシステム[5-11]では、楽曲の表層的な構造を直接操作できないユーザでも、タイムスパン木を用いて楽曲を操作することを可能としていた。しかしそこで用いたタイムスパン木は、音楽家が手作業で作成したもので、その自動獲得は実現されていなかった。

GTTM を計算機上に実装する際の問題点は、楽曲の解釈自体が暗黙で主観的なものであるため、GTTM を含めた音楽理論も必然的に曖昧さを持っていることである。それに加え、多くの音楽理論は人間に説明することを目的としており、計算機上への実装が考慮されていない点も問題である。従来、GTTM の計算機上への実装が試みられたが[12][13]、それらの手法ではルール競合の問題が解消できていなかった。一方、文献[14]ではタイムスパン木の獲得を可能としていたが、手作業によるルールの適用が必須であった。したがって、これらの手法[12-14]では分析を自動化することはできなかった。

そこで本研究では、GTTMのルールの持つ曖昧性の問題を解消し、分析の自動化を実現するため、計算機上で実行可能となるよう理論を拡張した exGTTM¹を提案する。その際、楽曲解釈の暗黙性、曖昧性、主観性をコントロールするための調節可能なパラメータを導入する。そして、exGTTMに基づきタイムスパン木を自動で分析するシステムATTA (Automatic Time-span Tree Analyzer)をPerl上に実装した。実験の結果、ユーザがパラメータを適切に設定することにより性能が向上することが確認できた。

2. GTTM の導入とその問題点

GTTM は、音楽に関して専門知識のある聴取者の直観を形式的に記述するための理論で、グルーピング構造分析、拍節構造分析、タイムスパン簡約、プロロンゲーション簡約という4つのサブ理論から構成されている。このうち本研究ではタイムスパン木の獲得に必要な、グルーピング構造分析、拍節構造分析、タイムスパン簡約を扱う。(図1)。

グルーピング構造分析は、連続したメロディをフレーズやモチーフなどに階層的に分割するも



図1：グルーピング構造、拍節構造、タイムスパン木

¹ exGTTMは、extended-GTTM (拡張したGTTM)と executable-GTTM (実行可能なGTTM)の両方の意味を併せ持つ。

ので、長いメロディを歌うときにどこで息継ぎすべきかを見つけるような分析である。拍節構造分析は、4分音符/2分音符/1小節/2小節/4小節など各拍節レベルにおける強拍と弱拍を同定するもので、聴取者が曲に合わせて手拍子を打つタイミングや指揮者がタクトを振るタイミングを求めるような分析である。タイムスパン簡約は、メロディの重要な部分と装飾的な部分を分離するもので、構造的に重要な音が幹になるような2分木(タイムスパン木)を求める分析である。図2の左図は、メロディとそのタイムスパン木を描いたものであるが、そのタイムスパン (<--->で表された部分)は、右図のようにヘッドと呼ばれる一つの音で代表させることができる(ここではC4の音)。

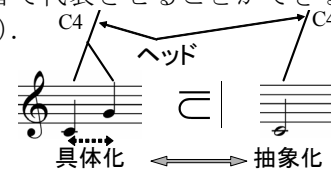


図2：メロディの包摂関係

GTTM の各サブ理論は、構成ルールと選好ルールの2種類からなる。構成ルールは、構造が成立するために必要な条件や制約で、選好ルールは、複数の構造が構成ルールを満たす場合どれが好ましいかを示すルールである。本節では、GTTMのルールを計算機上に実装する上での問題点について述べる。

2.1 選好ルールの定義の曖昧さ

GTTM は、ルールの定義に未定義の用語を用いており、それが分析に曖昧さをもたらしている。たとえば、GTTMには類似したメロディをひとまとまりの構造として選択しようとするルールがあるが、メロディの類似度については未定義である。

この問題の解決法として、本研究では各ルールが成立するかどうかを決める基準を形式化する。

2.2 選好ルールの競合

選好ルールを適用する際、ルールの適用順序が決まっていないため、ルールの競合がしばしば起きる。図3は、グルーピング選好ルール(GPRs: Grouping Preference Rules)が競合している例である。図3では、GPR3aが第3音-4音間に適用され、GPR6が第5音-6音間に適用されているが、GPR1は第4音が単音のグループになることを嫌うため、両者を境界とすることはできない。そこでルールの競合が生じる。

本研究では、ルールの優先順位を制御するためのパラメータを導入することで、この問題を解決する。



図3：ルールの競合の例

2.3 有効なアルゴリズムが定義されていない

GTTM の選好ルールは、複数の構造のうちどれが好ましいかを示しているだけで、階層的な構造を獲得するために必要な手順が定義されていない。

GTTM で述べられているように、構成ルールを満たすすべての構造を用意した後に、選好ルールを用いてその中から最適な構造を選択するという手法は、現実的ではない。たとえば、たった 10 音の音符列から 185794560 ($=9^2 \times 9!$) 種類ものタイムスパン木が得られるため、その中から最適な構造を選択することは容易でない。

この問題を解決するため、本研究では GTTM の本[1]にある数少ない適用例を参考に、階層的な構造を獲得するためのアルゴリズムを構築した。

2.4 フィードバックリンクに関する説明の不足

GTTM では、高次の構造から低次の構造へのフィードバックリンクがある。たとえば、GPR7 では、「タイムスパン簡約やプロロンゲーション簡約が安定するグルーピング構造を選択する」と定義されている。しかし、それらのルールに関する詳細な説明がなく、またルールの適用例も少ないため実装が困難である。

3. exGTTM : 計算機上で実行可能な GTTM

計算機上で実行可能となるよう理論を拡張した exGTTM を提案する。exGTTM は、タイムスパン木を獲得するための 36 個のルールのうち現在 26 個を扱っている。残る 10 個のルールのうち 4 つはフィードバックリンクに関するもので、6 つは和音に関するものである。本研究では、システムの性能を正しく評価するため、分析の対象を和音を含まない単旋律に限定する。

本節では、計算機上に実装するために GTTM に加えた拡張について説明する。拡張の方針については、グルーピング構造分析、拍節構造分析、タイムスパン簡約の 3 つの分析において共通している。

3.1 ルールの再形式化

各選好ルールを数式化することによって、複数のルールの適用結果の統合が容易に行えるようにする。 D_i^{rule} はルールが成立する度合いを表す変数で、ルールが成立する場合 1、成立しない場合 0 を表す関数である。

成立するかどうかは明らかなルールの場合、 D_i^{rule} は 1 か 0 の 2 値を出力する。たとえば、GPR2b は、発音時刻間隔が前後の発音時刻間隔と比べて長い場合に成立するルールで、 D_i^{GPR2b} の値は、成立する場合 1、成立しない場合 0 を示す。

$$D_i^{GPR2b} = \begin{cases} 1 & ioi_{i-1} < ioi_i \text{ and } ioi_i > ioi_{i+1} \\ 0 & \text{else} \end{cases} \quad (1)$$

ただし、
 i : i 番目の遷移 (i 音目と $i+1$ 音目の間)
 ioi_i : 発音時刻間隔

一方、成立するかどうか曖昧なルールの場合、 D_i^{rule} は 1 から 0 までの値を出力する。たとえば、TSRPR3 (Attack-points) は、メロディ中の音高の高い音をタイムスパンのヘッドとして選択することを好むルールで、 D_i^{TSRPR3} は以下のように音高が高いほど大きな値を示す。

$$D_i^{TSRPR3} = \text{pitch}_i / \max_j \text{pitch}_j \quad (2)$$

ただし、
 pitch_i : i 音目のヘッドの音の音高

3.2 未定義の概念の詳細化

2.1 節で述べたように、GTTM では、未定義の用語をルールの定義に用いており、それが分析の曖昧さを招いている。

GTTM 本の著者 Lerdhal と Jackendoff が未定義な用語に詳細な定義を加えることを避けた理由は、その用語の概念自体が曖昧で、唯一の定義がないためであると思われる。たとえば、メロディの類似度については、妥当と思われる多くの定義[15]があるが、最も適切なものを選ぶことは困難である。

そこで本研究では、我々が重要だと考える以下の 2 つの方針に基づき、未定義の概念に詳細な定義を与えることを試みた。

- 1) 直観的でわかりやすい定義であること
- 2) 調節可能なパラメータを導入し、曖昧さをできる限り排除すること

3.2.1 シンメトリーの定義

GPR5 は、シンメトリーに関するグルーピング選好ルールで、グループの分割が長さの等しい 2 つの部分からなるようグルーピングすることを優先する。しかし、どのような場合がシンメトリーであるかについては定義されていない。

そこで本研究では、分割されたグループの長さが等しいほど高い値を示す関数 D_i^{GPR5} を定義し、その関数によってシンメトリーである度合いを表現することにする。ここでは、そのような関数として平均をグループの中心、分散を σ とする正規分布を用いる。ここで、正規分布を採用した理由は、 σ を調節可能なパラメータとすることで、シンメトリーな度合い D_i^{GPR5} の形状を自由に調節できるからである。

$$D_i^{GPR5} = \frac{1}{\sqrt{2\pi\sigma}} e^{-\left\{ \frac{\sum_{j=start}^i ioi_j - \sum_{j=start}^{end} ioi_j / 2}{2} \right\}^2 / 2\sigma^2} \quad (3)$$

ただし、
 $start$: グループの始まりの遷移
 end : グループの終わりの遷移

図 4aは、グルーピングレベルaに対応したシンメトリーな度合い D_i^{GPR5} である。すべてのグルーピング選好ルールが適用された後、次のレベルのグルーピング構造がグルーピングレベルbのようになった場合、シンメトリーな度合い D_i^{GPR5} は図 4bのようになる。

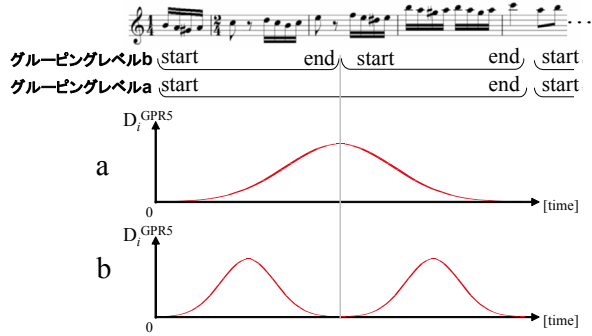


図 4：シンメトリーな度合い D_i^{GPR5}

3.2.1 パラレリズムの定義

GPR6 とMPR1, TSRPR4 はパラレリズムに関するルールで並列性のある部分は、並列的な構造を優先する。それぞれ対象とする構造が異なるため、3つのルールそれぞれに対して、パラレルな度合い $D_i^{(GPR6/MPR1/TSRPR4)}$ を定義する。ここでは紙面の都合上GPR6についてのみ説明する。

D_i^{GPR6} は、パラレルな部分の開始部と終端部で値が大きくなるような関数である (図 5)。GPR6では、調整可能な3つのパラメータ W_s, W_r, W_l を導入する。 W_s は、リズム方向のずれと音高のずれのどちらを重視するか決めるパラメータ、 W_r は、パラレルな区間の始まりと終わりどちらを重視するかを決めるパラメータ、 W_l は、パラレル区間の長さをどのくらい重視するかを決めるパラメータである。これらのパラメータを調節することによって、ユーザは簡単にパラレルなセグメントを見つけ、パラレル区間を設定することができる。

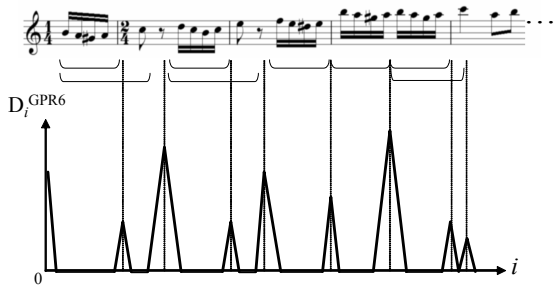


図 5：パラレルな度合い D_i^{GPR6}

3.3 優先順位の付与による競合の解消

GTTMの各選好ルールの強さを調節するためのパラメータ S^{rule} を導入する。このパラメータを導入することによって、局所的な、グルーピング境界/

拍点/ヘッド、の強さを求めることができようになる。たとえば、局所的なグルーピングルールを適用した結果求まる D_i^{GPR} と調節可能なパラメータ S_i^{rule} の重み付け和を求めることによって次式のように局所的なグルーピング境界の強さが求まる。

$$B_i = \frac{\sum_{j=(2a,2b,3a,3d,6)} D_i^{GPRj} \times S^{GPRj}}{\max_{i'} \left(\sum_{j=(2a,2b,3a,3b,3c,3d,6)} D_i^{GPRj} \times S^{GPRj} \right)} \quad (4)$$

3.4 階層構造獲得のためのアルゴリズム

階層構造を獲得するためのトップダウンなアルゴリズムを構築する。階層的な構造は、ボトムアップによる局所的な、グルーピング境界/拍点/ヘッド、の強さの算出とトップダウンによる次の構造の選択を繰り返すことによって獲得できる。

● グルーピング構造の獲得

- (1) 楽曲全体を一つのグループとする。
- (2) 局所的な境界の強さを算出し、局所的境界を検出する。
- (3) 最も強い境界でグループを2つに分割する。
- (4) グループ中に局所的境界がある限り(3)を繰り返す。

● 拍節構造の獲得

- (1) すべての拍点を最も低次の拍節構造とする。
- (2) 局所的な拍点の強さを算出する。
- (3) 可能な拍節構造の中から、次の階層の拍節構造を選択する。
- (4) 現在の拍節構造に2つ以上の拍点がある限り(2)、(3)を繰り返す。

● タイムスパン木の獲得

- (1) すべての音符をヘッドとする。
- (2) 局所的なヘッドの強さを算出する。
- (3) 各タイムスパンから次の階層のヘッドを選択する。
- (4) タイムスパンに2つ以上のヘッドがある限り(2)、(3)を繰り返す。

4. ATTA：タイムスパン木分析システム

本節では、以上述べてきたような exGTTM を計算以上に実装したシステム ATTM について述べる。ATTA は、グルーピング構造分析器、拍節構造分析器、タイムスパン木分析器の3つからなる (図 6)。それぞれの詳細については、付録に記述する。ここでは、ATTA が持つ3つの特長について述べる。

4.1 XML に基づくデータ構造

ATTA の入出力データには XML を採用した。ATTA の各分析器は独立に動作するが、XML に基づくデータ構造によりお互いに結びついている。システムの入力形式には、楽譜作成や分析、検索ツールが普及しており、フォーマットの相互変換が容易な MusicXML[16] を採用した。そして、システムの入力形式として、GroupingXML、

MetricalXML, Time-SpanXML を提案する。XML に基づくデータ構造は、階層的なグルーピング構造、拍節構造、タイムスパン木を表現する上で極めて適している。また、GroupingXML, MetricalXML, Time-SpanXML 上のノートエレメントは Xpointer[17]と Xlink[18]によってお互いに結合している。

我々は, MusicXML と共にグルーピング構造, 拍節構造, タイムスパン木を同時に配布することによって, 検索や編曲など様々な音楽タスクに利用されることを期待している。

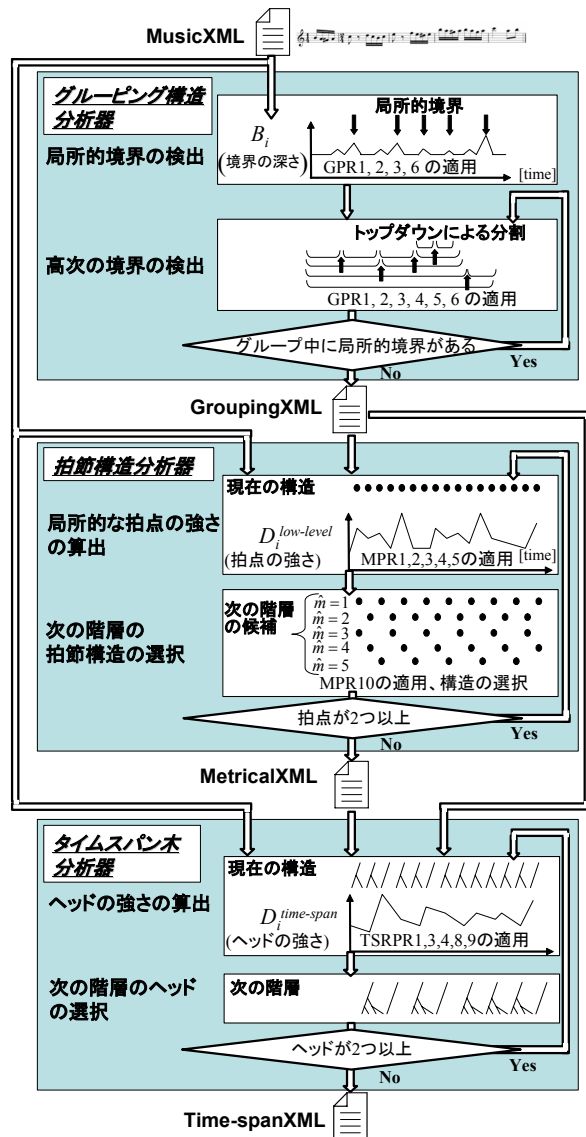


図 6 : ATTA の処理の流れ

4.2 Perl 上への実装

CGI を用いてインターネット越しに ATTA を使用できるよう Perl 上に実装した。我々は, このようなリソースの配布は音楽研究者のコミュニティにとって極めて重要であると考えている。

現在, <http://staff.aist.go.jp/m.hamanaka/atta/>にて ATTA を公開している。ATTA はタイムスパン木の自動獲得を実現した初めてのシステムであり, 今後開発されるシステムとベンチマークされることを期待している。

図 7 は, グルーピング構造分析器, 拍節構造分析器, タイムスパン木分析器のスクリーンショットである。それぞれの左側がパラメータを調節するためのスクロールバーで, 右側が分析結果を表示するための画面である。

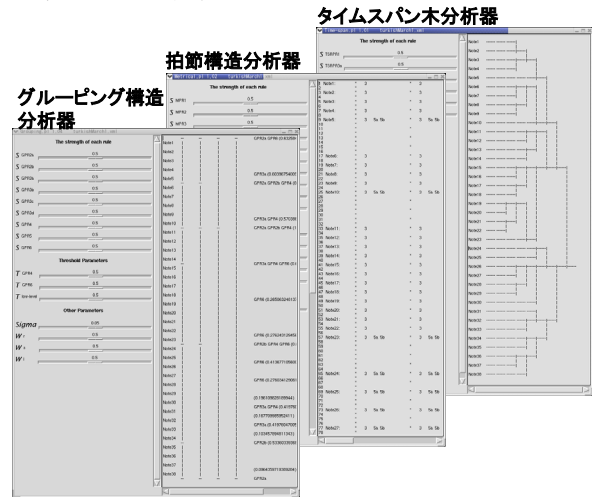


図 7 : ATTA のスクリーンショット

4.3 Java ベースの GUI

Perl 上に実装した分析器もシンプルなユーザーインターフェイスを備えているが, さらに Java ベースのグラフィックインターフェイス (GUI) を構築した。Java ベースの GUI は, 自動分析モードと手作業編集モードの 2 つのモードを備えている。自動分析モードでは, ATTA による自動分析を行いその結果を表示する。手作業編集モードでは, グルーピング構造, 拍節構造, タイムスパン木の編集を補助する。自動分析モードの結果を手作業編集モードで編集することもできる。

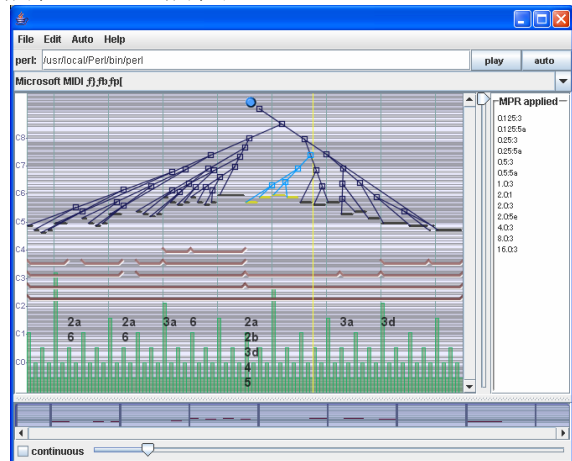


図 7 : Java ベースの GUI

5. 実験結果

本節では、ATTA の性能の評価を、適合率 P (precision)と再現率 R (recall)を組み合わせた F 値で評価する。F 値は、適合率と再現率の値が高いほど高くなる。

$$F_{\text{値}} = 2 \times \frac{P \times R}{P + R} \quad (5)$$

ただし、

P : 適合率 (正解データと同じグループ/拍点/ヘッドが、システムの出力に含まれている割合)

R : 再現率 (システムの出力と同じグループ/拍点/ヘッドが、正解データに含まれている割合)

このような F 値による評価を行うためにはグルーピング構造分析、拍節構造分析、タイムスパン簡約の正解データを用意する必要がある。正解データは、GTTM をよく理解している一人の音楽家が、クラシック曲から切り出した 8 小節の長さの 100 個のメロディに対してグルーピング構造分析、拍節構造分析、タイムスパン簡約を行ったものを、3 人の GTTM の専門家がクロスチェックしたものである。

分析結果は、パラメータの調整によって変化する(表 1)。そこで、まずパラメータ調節前 (ベースライン)の性能を求める。パラメータの初期値は、 $S_{rules}=0.5$, $T_{rules}=0.5$, $\sigma=0.05$, $W_s=0.5$ $W_r=0.5$, $W_l=0.5$ である。そして、手作業で 1 曲につき 10 分間でパラメータの調整を行った結果、F 値がベースラインより向上していることが確認できた (表 2)。

6. まとめ

本研究では、GTTM に基づきタイムスパン木を自動で獲得するシステム ATTA を開発した。本研究の主な意義は以下の 3 点である。

- exGTTM を提案した
計算機上で実行可能となるよう音楽理論 GTTM を拡張した exGTTM を提案した。GTTM の実装の困難さについてはこれまでたびたび指摘されてきたが[19]、その根本的な解決法は提案されてこなかった。本研究では、GTTM のルールを再形式化する際に、調節可能なパラメータを用いてルールを数式化することで、ルールの定義と分析の曖昧さとを分離することを可能にした。
- ATTA を計算機上に実装した
実際に計算機上で動作するシステム ATTA を実現し、階層的なグルーピング構造、拍節構造、タイムスパン木の自動獲得を可能にした。ATTA は現在 Web 上で CGI アプリケーションとして公開しており、今後開発されるシステムとベンチマークすることが可能である。
- 正解データを作成し実験した
パラメータの調節の結果、分析器の性能が向上することが確認できた。実験用に作成した、100 曲の正解データは、GTTM の分析結果のデータベースとしては最も多くの曲数を収めている。今後順次公開していく予定である。

表 1. 調節可能なパラメータ

	パラメータ	説明
グルーピング構造分析器	S^{GPRj}	各グルーピング選好ルールの強さ. $j=(2a, 2b, 3a, 3b, 3c, 3d, 4, 5, 6)$ $0 \leq S^{GPRj} \leq 1$
	σ	GPR5 で用いる正規分布の分散. $0 \leq \sigma \leq 0.1$
	W_s	GPR6 でパラレルな区間の始まりと終わりでどちらを重視するか決めるパラメータ. $0 \leq W_s \leq 1$
	W_r	GPR6 でリズム方向のずれと音高方向のずれのどちらを重視するか決めるパラメータ. $0 \leq W_r \leq 1$
	W_l	GPR6 でパラレルな区間の長さをどのぐらい重視するか決めるパラメータ. $0 \leq W_l \leq 1$
	T^{GPR4}	GPR4 で GPR2,3 の効果が明白であるかを決定するパラメータ. $0 \leq T^{GPR4} \leq 1$
	$T^{low-level}$	局所的な境界であるかを決定する閾値. $0 \leq T^{low-level} \leq 1$
拍節構造分析器	S^{MPRj}	各拍節選好ルールの強さ. $j=(1, 2, 3, 4, 5a, 5b, 5c, 5d, 5e, 10)$ $0 \leq S^{MPRj} \leq 1$
	W_r	MPR1 でリズム方向のずれと音高方向のずれのどちらを重視するか決めるパラメータ. $0 \leq W_r \leq 1$
	T^{MPRj}	各ルールが成立するかを決定する閾値. $j=(4, 5a, 5b, 5c)$ $0 \leq T^{MPRj} \leq 1$
タイムスパン木分析器	S^{TSRPRj}	各タイムスパン簡約選好ルールの強さ. $j=(1, 3a, 3b, 4, 8, 9)$ $0 \leq T^{MPRj} \leq 1$

表 2. パラメータ調節前と後の F 値

メロディ	グルーピング構造分析		拍節構造分析		タイムスパン簡約	
	ベースライン	調節後	ベースライン	調節後	ベースライン	調節後
1. 楽興の時	0.18	0.56	0.95	1.00	0.71	0.84
2. 子守歌	0.76	1.00	0.83	0.85	0.54	0.69
3. トロイメライ	0.60	0.87	0.76	1.00	0.50	0.63
4. よろこびの歌	0.12	0.73	0.95	1.00	0.22	0.48
5. 舟唄	0.04	0.54	0.72	0.79	0.24	0.60
	⋮	⋮	⋮	⋮	⋮	⋮
平均(100 個のメロディ)	0.46	0.77	0.84	0.90	0.44	0.60

参考文献

- [1] Lerdahl, F., and R. Jackendoff. A Generative Theory of Tonal Music. Cambridge, Massachusetts: MIT Press, 1983.
- [2] G. Cooper, and L. B. Meyer. The Rhythmic Structure of Music. The University of Chicago Press, 1960.
- [3] E. Narmour. The Analysis and Cognition of Basic Melodic Structure. The University of Chicago Press, 1990.
- [4] D. Temperley. The Cognition of Basic Musical Structures. MIT press, Cambridge, 2001.
- [5] 平田圭二, 青柳龍也, 音楽理論 GTTM に基づく多声音楽の表現手法と基本演算, 情報処理学会論文誌 Vol.43, No.2, 2002.
- [6] K. Hirata, and T. Aoyagi. "Computational Music Representation on the Generative Theory of Tonal Music and the Deductive Object-Oriented Database." Computer Music Journal 27(3), 73-89, 2003.
- [7] N. Todd. A Model of Expressive Timing in Tonal Music. Musical Perception, 3:1, 33-58, 1985.
- [8] Widmer, G. "Understanding and Learning Musical Expression", Proceedings of International Computer Music Conference, pp. 268-275, 1993.
- [9] Hirata, K., and Hiraga, R. "Ha-Hi-Hun plays Chopin's Etude", Working Notes of IJCAI-03 Workshop on Methods for Automatic Music Performance and their Applications in a Public Rendering Contest, pp. 72-73, 2003.
- [10] 平田圭二, 松田周, パピプーーン: GTTM に基づく音楽要約システム, 情報処理学会研究報告 2002-MUS-46, pp.29-36, 2002.
- [11] K.Hirata, and S. Matsuda. "Interactive Music Summarization based on Generative Theory of Tonal Music." Journal of New Music Research, 32:2, 165-177, 2003.
- [12] 井田健太郎, 平田圭二, 東条敏, GTTM に基づくグルーピング構造および拍節構造の自動分析の試み, 情報処理学会研究報告 2001-MUS-42, pp.49-54, 2001.
- [13] 東洋武士, 平田圭二, 東条敏, 佐藤健, グルーピング規則適用を改良した GTTM の実装, 情報処理学会研究報告 2002-MUS-47, pp.121-126, 2002.
- [14] T. Nord. A. Toward Theoretical Verification: Developing a Computer Model of Lerdahl and Jackendoff's Generative Theory of Tonal Music. Ph.D. Thesis, The University of Wisconsin, Madison, 1992.
- [15] Hewlett, W. B. ed. Melodic Similarity: Concepts, Procedures, and Application. Computing in Musicology 11, The MIT press, Cambridge, 1998.
- [16] Recordare LLC. "MusicXML 1.0 Tutorial." <http://www.recordare.com/xml/musicxml-tutorial.pdf>, 2004.
- [17] W3C. "XML Pointer Language (XPointer)." <http://www.w3.org/TR/xptr/>, 2002.
- [18] W3C. "XML Linking Language (XLink) Version 1.0." <http://www.w3.org/TR/xlink/>, 2001.
- [19] V. Heikki. "Lerdahl and Jackendoff Revisited." http://www.cc.jyu.fi/~heivalko/articles/lehr_jack.htm.

付録 1 グルーピング構造分析器

• ステップ 1: 基本変数の算出.

MusicXMLから 6 つの基本変数を算出する. 6 つの変数はそれぞれ, 消音時刻から次の発音時刻までの間隔 $rest_i$, 発音時刻間隔 ioi_i , 音高の差 $regi_i$, ダイナミクスの差 dyn_i , 楽譜上の音符の長さとして実際に演奏された音の長さの比の差 $arti_i$, 音価の差 len_i である. 各基本変数について添え字の i が何番目の遷移であるかを表している. ただし, 第 1 音から第 2 音への変化を遷移 1, 第 2 音から第 3 音への変化を遷移 2 とする.

• ステップ 2: GPR の適用.

$$D_i^{GPR1} = \begin{cases} 1 & B_{i-1} \leq B_i \text{ and } B_i \geq B_{i+1} \\ 0 & \text{else} \end{cases} \quad (6)$$

$$D_i^{GPR2a} = \begin{cases} 1 & rest_{i-1} < rest_i \text{ and } rest_i > rest_{i+1} \\ 0 & \text{else} \end{cases} \quad (7)$$

$$D_i^{GPR2b} = \begin{cases} 1 & ioi_{i-1} < ioi_i \text{ and } ioi_i > ioi_{i+1} \\ 0 & \text{else} \end{cases} \quad (8)$$

$$D_i^{GPR3a} = \begin{cases} 1 & regi_{i-1} < regi_i \text{ and } regi_i > regi_{i+1} \\ 0 & \text{else} \end{cases} \quad (9)$$

$$D_i^{GPR3b} = \begin{cases} 1 & dyn_{i-1} = 0 \text{ and } dyn_i \neq 0 \text{ and } dyn_{i+1} = 0 \\ 0 & \text{else} \end{cases} \quad (10)$$

$$D_i^{GPR3c} = \begin{cases} 1 & arti_{i-1} = 0 \text{ and } arti_i \neq 0 \text{ and } arti_{i+1} = 0 \\ 0 & \text{else} \end{cases} \quad (11)$$

$$D_i^{GPR3d} = \begin{cases} 1 & len_{i-1} = 0 \text{ and } len_i \neq 0 \text{ and } len_{i+1} = 0 \\ 0 & \text{else} \end{cases} \quad (12)$$

$$D_i^{GPR4} = \begin{cases} 1 & \max(P_i^{rest}, P_i^{ioi}, P_i^{regist}, P_i^{dyn}, P_i^{arti}) > T^{GPR4} \\ 0 & \text{else} \end{cases} \quad (13)$$

$$D_i^{GPR5} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{\sum_{j=start}^i ioi_j - \sum_{j=start}^{end} ioi_j / 2}{\sigma}\right)^2 / 2\sigma^2} \quad (14)$$

$$D_i^{GPR6} = \sum_j \sum_r \begin{cases} G_{ijr}^{start} \times W_s' & m_{ij} = b \\ G_{ijr}^{end} \times W_s & m_{ij} = e \\ G_{ijr}^{start} \times W_s' + G_{ijr}^{end} \times W_s & m_{ij} = t \\ 0 & m_{ij} = s \end{cases} \quad (15)$$

ただし,

$$B_i = \sum_{j=(2a,2b,3a,3d,6)} D_i^{GPRj} \times S^{GPRj} / \max_{i'} \left(\sum_{j=(2a,2b,3a,3b,3c,3d,6)} D_i^{GPRj} \times S^{GPRj} \right)$$

$$P_i^{rest} = rest_i / \sum_{j=i-1}^{i+1} rest_j, P_i^{ioi} = ioi_i / \sum_{j=i-1}^{i+1} ioi_j, P_i^{dyn} = dyn_i / \sum_{j=i-1}^{i+1} dyn_j,$$

$$P_i^{regist} = \begin{cases} regi_i / \sum_{j=i-1}^{i+1} regi_j & \left(\sum_{j=i-1}^{i+1} regi_j > 0 \right) \\ 0 & \text{else} \end{cases}, P_i^{arti} = arti_i / \sum_{j=i-1}^{i+1} arti_j$$

$start$: グループの始まりの遷移

end : グループの終わりの遷移

div : 4 分音符の音価

r : パラレル部分の長さ (一拍の整数倍)

$$q_i = \left\lfloor \frac{\sum_{k=1}^i ioi_k}{div} \right\rfloor \left(\left[\lfloor \cdot \rfloor \right] \text{はガウス記号. } [n] \text{は} n \text{を越えない最大の整数を表す.} \right)$$

$$G_{ijr}^{start} = \frac{z_{q_i q_j r}}{y_{q_i q_j r}} \times W_r' \times r^{W_l} + \frac{y_{q_i q_j r}}{x_{q_i r} + x_{q_j r}} \times W_r \times r^{W_l}$$

$$G_{ijr}^{end} = \frac{z_{q_i-r q_j-r r}}{y_{q_i-r q_j-r r}} \times W_r' \times r^{W_l} + \frac{y_{q_i-r q_j-r r}}{x_{q_i-r r} + x_{q_j-r r}} \times W_r \times r^{W_l}$$

$$W_r' = 1 - W_r, W_s' = 1 - W_s, W_l' = 1 + W_l$$

$$x_{q_i r} = \sum_j \begin{cases} 1 & q_i \leq q_j \text{ and } q_j \leq q_i + r \\ 0 & q_i > q_j \text{ or } q_j > q_i + r \end{cases}$$

$$y_{q_i q_j r} = \sum_k \sum_l \begin{cases} 1 & (q_i - q_j) \times div = \sum_{g=1}^k ioi_g - \sum_{g=1}^l ioi_g \\ 0 & (q_i - q_j) \times div \neq \sum_{g=1}^k ioi_g - \sum_{g=1}^l ioi_g \end{cases}$$

$$z_{q_i q_j r} = \sum_k \sum_l \begin{cases} 1 & (q_i - q_j) \times \text{div} = \sum_{g=1}^k \text{ioi}_{i_g} - \sum_{g=1}^l \text{ioi}_{j_g} \text{ and } \text{regi}_i = \text{regi}_j \\ 0 & (q_i - q_j) \times \text{div} \neq \sum_{g=1}^k \text{ioi}_{i_g} - \sum_{g=1}^l \text{ioi}_{j_g} \text{ or } \text{regi}_i \neq \text{regi}_j \end{cases}$$

$$m_{i,j} = \begin{cases} b & q_i \neq q_{i-1} \text{ and } q_j \neq q_{j-1} \text{ and } q_i = q_{i+1} \text{ and } q_j = q_{j+1} \\ e & q_i = q_{i-1} \text{ and } q_j = q_{j-1} \text{ and } q_i \neq q_{i+1} \text{ and } q_j \neq q_{j+1} \\ t & q_i \neq q_{i-1} \text{ and } q_j \neq q_{j-1} \text{ and } q_i \neq q_{i+1} \text{ and } q_j \neq q_{j+1} \\ s & \text{else} \end{cases}$$

• ステップ 3： 局所的境界の検出.

次式により遷移*i*がグループの境界となる($D_i^{\text{low-level}}=1$)かそうでない($D_i^{\text{low-level}}=0$)か求まる.

$$D_i^{\text{low-level boundary}} = \begin{cases} 1 & B_i > T^{\text{low-level}} \text{ and } D_i^{\text{GPR1}} = 1 \\ 0 & \text{else} \end{cases} \quad (16)$$

• ステップ 4： 階層的なグルーピング構造の獲得

グループがその内部に局所的境界を含んでいる場合、次式によって次の階層の境界*i*が再帰的に求まる.

$$\hat{i} = \underset{i}{\text{argmax}} (D_i^{\text{low-level boundary}} \times \sum_j D_j^{\text{GPR1}} \times S^{\text{GPR1}}) \quad (17)$$

ただし、

$$j = (2a, 2b, 3a, 3d, 4, 5, 6)$$

i はグループに含まれるすべての遷移

付録 2 拍節構造分析器

• ステップ 1： 基本変数の算出.

MusicXMLから 5 つの基本変数を算出する. 5 つの変数はそれぞれ、拍点から始まる音のベロシティ $velo_i$, 音価 $valu_i$, 連続する音量の長さ vol_i , スラーの長さ $slur_i$, 音高 num_i である. 各基本変数について添え字の*i*は、現在の拍節レベルで何番目の拍点であるかを表している. 5 つの基本変数の平均値をそれぞれ μ_{velo} , μ_{valu} , μ_{vol} , μ_{slur} , μ_{num} とする.

• ステップ 2： MPR の適用.

$$D_{ik}^{\text{MPR1}} = \begin{cases} 1 & \frac{y_{ik}}{x_{ik}} \times W_r + \frac{z_{ik}}{y_{ik}} \times (1 - W_r) > T^{\text{MPR1}} \\ 0 & \text{else} \end{cases} \quad (18)$$

$$D_i^{\text{MPR2}} = \begin{cases} 1 & (i^{\text{end}} - i) / (i^{\text{end}} - i^{\text{start}}) \\ 0 & \text{else} \end{cases} \quad (19)$$

$$D_i^{\text{MPR3}} = \begin{cases} 1 & velo_i > 0 \\ 0 & velo_i = 0 \end{cases} \quad (20)$$

$$D_i^{\text{MPR4}} = \begin{cases} 1 & velo_i > 2 \times \mu_{velo} \times T^{\text{MPR4}} \\ 0 & velo_i \leq 2 \times \mu_{velo} \times T^{\text{MPR4}} \end{cases} \quad (21)$$

$$D_i^{\text{MPR5a}} = \begin{cases} 1 & valu_i > 2 \times \mu_{valu} \times T^{\text{MPR5a}} \\ 0 & valu_i \leq 2 \times \mu_{valu} \times T^{\text{MPR5a}} \end{cases} \quad (22)$$

$$D_i^{\text{MPR5b}} = \begin{cases} 1 & vol_i > 2 \times \mu_{vol} \times T^{\text{MPR5b}} \\ 0 & vol_i \leq 2 \times \mu_{vol} \times T^{\text{MPR5b}} \end{cases} \quad (23)$$

$$D_i^{\text{MPR5c}} = \begin{cases} 1 & slur_i > 2 \times \mu_{slur} \times T^{\text{MPR5c}} \\ 0 & slur_i \leq 2 \times \mu_{slur} \times T^{\text{MPR5c}} \end{cases} \quad (24)$$

$$D_i^{\text{MPR5d}} = \begin{cases} 1 & D_i^{\text{MPR5a}} = 1 \text{ and } D_{i+1}^{\text{MPR5a}} = 1 \\ 0 & D_i^{\text{MPR5a}} \neq 1 \text{ or } D_{i+1}^{\text{MPR5a}} \neq 1 \end{cases} \quad (25)$$

$$D_i^{\text{MPR5e}} = \begin{cases} 1 & num_i = num_{i+1} \\ 0 & num_i \neq num_{i+1} \end{cases} \quad (26)$$

ただし、

i^{start} : グループの先頭の拍点の*i*

i^{end} : グループの終わりの拍点の*i*

$$x_{ik} = \sum_{i^{\text{start}}}^{i^{\text{end}}} \begin{cases} 1 & velo_i > 0 \\ 0 & velo_i = 0 \end{cases} + \sum_{k^{\text{start}}}^{k^{\text{end}}} \begin{cases} 1 & velo_k > 0 \\ 0 & velo_k = 0 \end{cases}$$

$$y_{ik} = \sum_{i^{\text{start}}}^{i^{\text{end}}} \begin{cases} 1 & velo_i > 0 \text{ and } velo_{k+i-1} > 0 \\ 0 & \text{else} \end{cases}$$

$$z_{ik} = \sum_{i^{\text{start}}}^{i^{\text{end}}} \begin{cases} 1 & velo_i > 0 \text{ and } num_{i-1} = num_i \text{ and } num_{k+i-1} = num_{k+i} \\ 0 & \text{else} \end{cases}$$

ステップ 3： 局所的な拍点の強さの算出.

次式により局所的な拍点の強さを算出する.

$$D_i^{\text{metrical}} = B_i + \sum_k \begin{cases} B_k \times S^{\text{MPR1}} & D_{ik}^{\text{MPR1}} = 1 \\ 0 & D_{ik}^{\text{MPR1}} = 0 \end{cases} \quad (27)$$

ただし、

$$B_i = \sum_{j=(2,3,4,5a,5b,5c,5d,5e)} D_j^{\text{MPR1}} \times S^{\text{MPR1}}$$

• ステップ 4： 階層的な拍節構造の獲得

現在の拍節レベルに拍点が 2 つ以上ある場合、次式によって次の階層の拍節構造 \hat{m} が求まる.

$$\hat{m} = \underset{m=(1,2,3,4,5)}{\text{argmax}} \sum_i \begin{cases} D_i^{\text{low-level metrical}} & (i-m) \bmod 2 = 0 \\ D_i^{\text{low-level metrical}} \times S^{\text{MPR10}} & (i-m) \bmod 3 = 1 \\ 0 & \text{else} \end{cases} \quad (28)$$

付録 3 タイムスパン木分析器

• ステップ 1： 基本変数の算出.

MusicXML, GroupingXML, Tims-SpanXML から 4 つの基本変数を算出する. 4 つの変数はそれぞれ、現在のヘッド*i*の音と次のヘッド*i+1*の音の、消音時刻から発音時刻までの間隔 $rest_i$, 発音時刻間隔 ioi_i , 音高の差 $pitch_i$, 現在のヘッド*i*の拍点の数 dot_i である.

• ステップ 2： TSRPR の適用.

$$D_i^{\text{TSRPR1}} = dot_i / \max_j dot_j \quad (29)$$

$$D_i^{\text{TSRPR3}} = pitch_i / \max_j pitch_j \quad (30)$$

$$D_i^{\text{TSRPR3}} = 1 - pitch_i / \max_j pitch_j \quad (31)$$

$$D_{ik}^{\text{TSRPR4}} = \begin{cases} 1 & ioi_{i-1} = ioi_{k-1}, ioi_i = ioi_k, ioi_{i+1} = ioi_{k+1} \\ 0 & \text{else} \end{cases} \quad (32)$$

$$D_i^{\text{TSRPR8}} = \begin{cases} 1 & i = i^{\text{start}} \\ 0 & \text{else} \end{cases} \quad (33)$$

$$D_i^{\text{TSRPR9}} = \begin{cases} 1 & i = i^{\text{end}} \\ 0 & \text{else} \end{cases} \quad (34)$$

• ステップ 3： 局所的なヘッドの強さの算出.

次式により局所的なヘッドの強さを算出する.

$$D_i^{\text{time-span}} = B_i + \sum_k \begin{cases} B_k \times S^{\text{TSRPR4}} & D_{ik}^{\text{TSRPR4}} = 1 \\ 0 & D_{ik}^{\text{TSRPR4}} = 0 \end{cases} \quad (35)$$

ただし、

$$B_i = \sum_{j=(1,3,8,9)} D_j^{\text{TSRPR1}} \times S^{\text{TSRPR1}}$$

• ステップ 4： 次の階層のヘッドの選択

現在のタイムスパンにヘッドが 2 つ以上ある場合、次式によって次の階層のヘッド \hat{h} が求まる.

$$\hat{h} = \begin{cases} i & D_i^{\text{time-span}} \geq D_j^{\text{time-span}} \\ j & \text{else} \end{cases} \quad (36)$$