音楽音響信号に対するビートトラッキングシステム

後藤 真孝　　村岡 洋一

早稲田大学 理工学部

{goto, muraoka}@muraoka.info.waseda.ac.jp

あらまし　本稿では，音楽の音響信号に対してリアルタイムにビートを認識するビートトラッキングシステムについて述べる．従来の研究の多くは，MIDI 信号か少数の楽器で演奏された音響信号を対象にしており，ドラムスを含む複数の楽器で演奏された音響信号を扱うことはできなかった．本システムでは，ロック・ポップスの主にドラムスがビートを刻む曲を対象とする．複数のエージェントが並列にビートを解釈することで，あるエージェントの解釈がはずれても，ビートを見失わずに正しい解釈を出力できる．また，主要なドラム音の出現位置を検出することにより，各ビートが強拍か弱拍かを判断する．並列計算機 AP1000 上に実装して実験した結果，30 曲中 27 曲に対してビートトラッキングできた．

A Real-time Beat Tracking System for Musical Acoustic Signals

Masataka Goto　　Yoichi Muraoka

School of Science and Engineering, Waseda University
3-4-1 Ohkubo Shinjuku-ku, Tokyo 169, JAPAN.

Abstract　This paper presents a beat tracking system that processes musical acoustic signals and recognizes temporal positions of beats in real time. Previous systems were not able to deal with acoustic signals that contained sounds of various instruments, especially drums. They dealt with either MIDI signals or acoustic signals played on a few instruments. Our system deals with popular music in which drums maintain the beat. Because our system examines multiple hypotheses in parallel, it can follow beats without losing track of them, even if some hypotheses become wrong. Our system has been implemented on a parallel computer, the Fujitsu AP1000. In our experiment, it correctly tracked beats in 27 out of 30 commercially distributed popular songs.

1　はじめに

ビートトラッキングとは，人間が音楽に合わせて手拍子を打つように，曲のビート (4 分音符) の位置を認識する技術である．人間が音楽に合わせて容易に手拍子を打てることからわかるように，ビートは西洋音楽を理解する上で基本的な概念の一つである．ビートトラッキングの実現は，音響信号に対する音楽聴取過程を計算機上で実現する第一段階として重要である．

従来のビートトラッキングの研究の多くは，MIDI 信号やそれに相当する入力を対象にしていた [1]−[7]．そのため入力には電子楽器しか用いることができず，適用範囲が制限されていた．これらの研究は，ピアノの独奏などのクラシック系の曲を対象にしたものが中心で，主にテンポ変化に追従することに重点を置いていた．一方，ソロや少数の楽器で演奏された音響信号を対象にした研究も報告されている [8] [9]．しかし，これらはドラムスを含む複数の楽器で演奏されたロック・ポップスのような音楽は扱えなかった．さらに自動採譜の一部として研究されていたため，リアルタイムに処理することはできなかった．

本研究では，音楽音響信号に対してリアルタイムにビートを認識・予測するビートトラッキングシステム (BTS) を実現する．BTS は，ロック・ポップスの主にドラムスがビートを刻む曲を対象とする．これらの曲ではクラシック系の曲に比べてテンポ変化が少ないため，テンポ変化に追従するよりはむしろ，音響信号中からいかにビートを見つけ出すかに研究の重点を置く．BTS は，市販の Compact Disc (CD) などから得た複数の楽器音を含む音響信号を入力とし，4 分音符に相当するビートを認識する．その際，ビートの存在する時刻を認識するだけでなく，そのビートが強拍か弱拍かも判断する．

音響信号に対するビートトラッキングを実現するには，主に次のような課題がある．(1) 対象とする音響信号には様々な楽器音が混在しており，MIDI の場合には容易に得られる各音の発音時刻を，正確に求めることは一般に不可能である．(2) 音響信号波形にはビートに直接関係のないエネルギーピークが多数あるため，ピーク検出した後に閾値処理をする単純な手法では，ビートをとらえることはできない．(3) ビートは音楽に対して人間が知覚する概念であり，ビートが実際の信号に直接対応するとは限らない．音響信号がないところにビートが存在する場合もある．このような曖昧さがあるためにビートを一意に解釈することはできず，常に複数の解釈の可能性がある．(4) 4 分音符の長さや，各ビートが強拍か弱拍かを判断することは，一般には難しい．

これらを解決するために，BTS ではビートの様々な解釈の可能性を並列に調べる．まず，周波数解析において各発音時刻に対して信頼度を計算することにより，高い信頼度の発音時刻をより重視して以後の処理をおこなう．次に，複数のエージェントが，これらの発音時刻をそれぞれ異なった戦略により解釈し，次のビートの時刻を予測する．最終的に出力するビートの時刻は，最も確信度の高いエージェントの解釈に基づいて決定する．さらに BTS は，ロック・ポップスの多くの曲で，バスドラムとスネアドラムがそれぞれ強拍と弱拍で多く鳴るという知識を利用する．これらのドラム音が存在すると仮定し，それらを検出することで，BTS は 4 分音符の長さを推定し，各ビートが強拍か弱拍かを判断できる．

計算量が多い処理をリアルタイムにおこなうために，富士通の分散メモリ型並列計算機 AP1000 上に BTS を実装した．BTS は，ビートの存在する時刻 (ビート時刻)，強拍か弱拍かの判断 (ビートタイプ)，現在のテンポ，の三つから成るビート情報をネットワーク上に出力する．市販の CD を用いて実験した結果，30 曲中 27 曲に対して正しいビート情報が得られた．

2　実現上の課題と解決法

音響信号に対してリアルタイムにビートトラッキングするシステムを実現するには，多くの課題がある．本節では，その中で主要な課題を，信号処理の課題，リアルタイム処理の課題，音楽的判断の課題の三つに分けて述べ，それぞれの解決法を示す．

□ 信号処理の課題

1. **ビートという特定の認識対象の信号が入力音中にあるわけではなく，音響信号がないところにビートが存在する場合もある．**

   BTS では，異なる周波数帯域ごとの発音時刻，主要なドラム音 (バスドラムとスネアドラム) の発音時刻，これらの音量，といった複数のビートの手がかりを抽出し，ビートを認識する．これらの手がかりを解釈するに当たり，ロック・ポップスの多くの曲におけるドラム音の鳴り方などの音楽的知識を用いる．

2. **MIDI の場合には容易に得られる各音の発音時刻を，音響信号中から正確には求められない．**

   周波数解析において，複数の異なる条件で発音時刻の信頼度付き候補を検出する．まず，周波数スペクトルから立ち上がり成分を抽出し，様々な周波数帯域や感度で発音時刻を求める．そして，こ

れらの発音時刻の信頼度を周波数成分の立ち上がりの度合から計算し，その信頼度を考慮して以後の処理をおこなう．

□ **リアルタイム処理の課題**

3. **リアルタイムに処理するため，過去に得られた音響信号しか利用できない．**

   過去に得られた発音時刻やビートから，次のビート時刻を事前に予測して出力する．ビートの存在する時刻の音響信号を認識し終ったときには，既にその発音時刻は過ぎてしまっているため，予測は不可欠である．

4. **現在のビートの解釈がはずれても，ビートを見失わずに正しい解釈に復帰する必要がある．**

   様々な解釈をする複数のエージェントが，並列に次のビート時刻を予測する．これにより，あるエージェントの解釈がはずれても，他のエージェントが正しく解釈している限り，ビートを見失わずにトラッキングできる．各エージェントはそれぞれ異なる戦略により発音時刻を解釈し，解釈の確信度を自己評価する．そして，最も確信度の高いエージェントの解釈に基づいてビート情報を生成する．

□ **音楽的判断の課題**

5. **各ビートが強拍か弱拍かを判断することは，何も制約のない状況では一般に難しい．**

   BTS ではこの判断をおこなうために，入力曲に以下の制約を設ける．

   - 入力曲は 4/4 拍子とする．
   - バスドラム (BD) が強拍 (1，3 拍目)，スネアドラム (SD) が弱拍 (2，4 拍目) で鳴る確率が高いものとする．

   後者は，すべての BD と SD が常にこの位置で鳴るという意味ではなく，単にその確率が高ければよいという制約である．これらの制約は，ロック・ポップスの多くの曲に当てはまる．

   BTS は，BD と SD の発音時刻を手がかりにして，ビートタイプ (強拍か弱拍か) を判断する．人間が音楽を聞いてビートトラッキングする際にも，BD と SD を大きな手がかりにしている．曲中の BD と SD の音が事前にはわからないので，ビートを認識しながらそれらの特徴周波数を自動的に獲得して検出をおこなう．ただし，BD と SD の検出は余分な誤検出が多く不安定なため，この結果だけからビートトラッキングすることはできない．そこで，ビート時刻を予測した後に，そのビートタイプを判断するためだけにこの検出結果を用いる．

6. **ビートの間隔が 4 分音符に相当するかどうか判断するのが難しい．**

   BTS では入力曲のテンポは，70〜180 M.M. (4 分音符/分) の間で，曲中を通じてほぼ一定であるものとする．実際に，ロック・ポップスではクラシック系の曲に比べてテンポ変化が少ない．しかし，たとえビートの間隔をこのテンポの範囲の 4 分音符の長さに限定しても，160 M.M. の曲を 80 M.M. と誤って判断するなど，4 分音符の長さを判断できない場合が生じる．

   BTS は，過去に認識したビート時刻に BD と SD がある程度交互に検出されていれば，その間隔を 4 分音符の長さと判断する．BD と SD が検出できない状況では，高い信頼度を持った発音時刻間の安定した間隔を，仮に 4 分音符の長さとみなす．

3 処理の概要

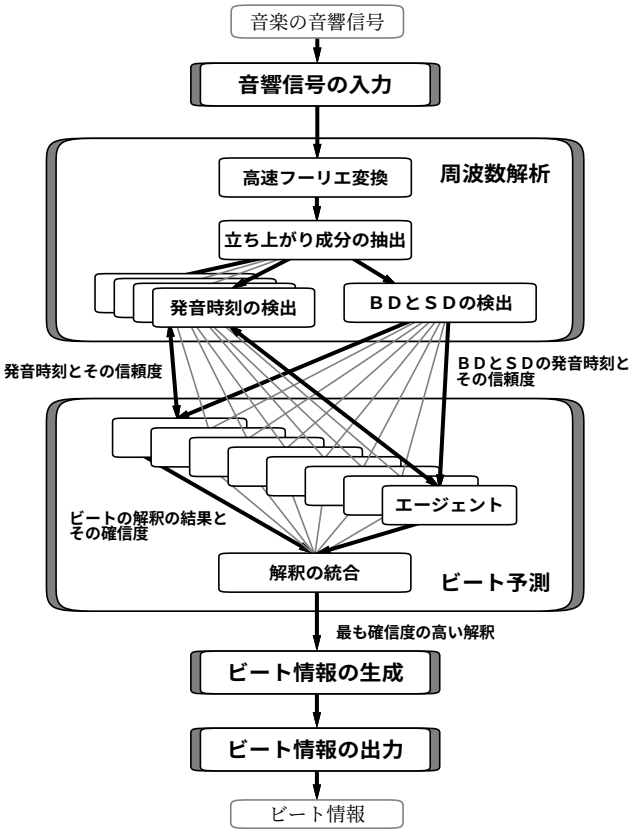

図 1: 処理の概要

BTS の処理の概要を図 1 に示す．まず，**音響信号の入力**において A/D 変換された音響信号に対して，**周波数解析**をおこない，各周波数帯域ごとの発音時刻と，BD と SD の発音時刻を検出する．次に，**ビート予測**

の複数のエージェントが，過去に得られた発音時刻から様々なビートの解釈を出力する．各エージェントはビートの間隔を求め，次のビート時刻の予測とビートタイプの判断をおこなった後に，解釈の確信度を自己評価する．そして，最も確信度の高い解釈の結果に基づいて，**ビート情報の生成**をおこなう．最後に**ビート情報の出力**が，ネットワークを通じて他のアプリケーションプログラムへとビート情報を送信する．

以下では，周波数解析とビート予測について詳しく述べる．

3.1 周波数解析

以下の処理により，異なる周波数帯域ごとの発音時刻と BD と SD の発音時刻が得られる．

3.1.1 高速フーリエ変換 (FFT)

A/D 変換された音響信号に対して FFT をおこない，周波数スペクトル (パワースペクトル) を得る．これにより，各周波数成分のパワーの時間変化が得られる．観測区間 (WinSIZE: WindowSize) を時間軸方向に単位時間 (ShiSIZE: ShiftSize) ずつずらしながら FFT を適用する．周波数軸方向の分解能[1] は WinSIZE によって決まり，時間軸方向の分解能[1] は ShiSIZE によって決まる．

3.1.2 立ち上がり成分の抽出

FFT の結果に対して周波数の立ち上がり成分を抽出し，それぞれの立ち上がりの度合を求める．式 (1) を満たす周波数成分 $p(t,f)$ を，立ち上がり成分とする (図 2)．

$$\begin{cases} p(t,f) & > & pp \\ np & > & pp \end{cases} \tag{1}$$

ただし，$p(t,f)$ は時刻 t，周波数 f におけるスペクトルのパワーとし，pp と np は 式 (2), 式 (3) で与えられるものとする．

$$pp = \max(p(t-1,f), p(t-1,f\pm1), p(t-2,f)) \tag{2}$$

$$np = \min(p(t+1,f), p(t+1,f\pm1)) \tag{3}$$

つまり，これによってパワーが増加し続けている周波数成分を抽出する．多くの場合，発音したときの鳴り始めの周波数成分がこの抽出で得られる．

立ち上がりの度合 $d(t,f)$ は，式 (4) により求める．

$$d(t,f) = p(t,f) - pp + \max(0, p(t+1,f) - p(t,f)) \tag{4}$$

[1] 現在の実装では，周波数分解能は 21.53Hz，時間分解能は 11.61msec である．

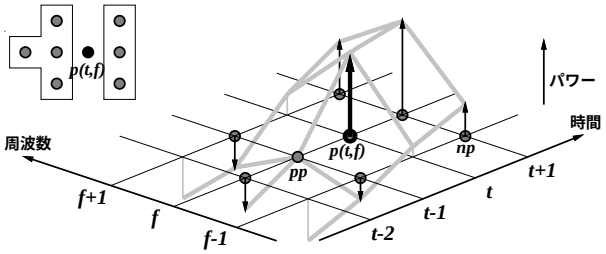

図 2: 立ち上がり成分の抽出

3.1.3 発音時刻の検出

各時刻における立ち上がり成分の合計値 $D(t)$ を式 (5) により求め，その時間変化のピーク時刻とピーク値を，平滑化微分を用いたピーク検出により求める[2]．

$$D(t) = \sum_{f} d(t,f) \tag{5}$$

こうして得られたピーク時刻を発音時刻とする．その信頼度は，$D(t)$ のピーク値の，最近得られた最大のピーク値に対する相対値とする．立ち上がりの度合が大きいほど信頼度が高くなる．

BTS では，複数のこのような発音時刻の検出器が，それぞれ異なるパラメータにより発音時刻を求める．各検出器は，**3.2** で述べるエージェントの組に対応しており，検出結果を対応する組へ送信する (図 1, 図 4)．それぞれの検出器は，検出感度，検出周波数帯域という二つのパラメータを持つ．検出感度は，平滑化微分における平滑化の時間幅であり，平滑化幅が小さいほど感度が高くなる．検出周波数帯域は，式 (5) 中の $\sum$ における周波数帯域の指定で，これにより周波数帯域ごとの発音時刻を別々に得ることができる．

3.1.4 BD と SD の検出

立ち上がり成分の抽出結果から，現在の曲の BD と SD の音に対応する特徴周波数を自動的に獲得する．そして，これらの特徴周波数を用いて BD と SD を検出する．まず，立ち上がり成分が発見された時刻において，周波数軸方向のピークを求め，そのヒストグラムを作成する (図 3)．その際，ヒストグラムには各ピークの立ち上がりの度合 $d(t,f)$ を加える．そして，ヒストグラム中で最も周波数の低いピークを BD，それより周波数が高く最もパワーの大きいピークを SD の特徴周波数とみなす．

立ち上がり成分が発見された時刻において，そのピーク周波数が BD か SD の特徴周波数に一致した場合，

[2] Savitzky と Golay の 2 次多項式適合による平滑化微分を用いたピーク検出により，ノイズの影響を平滑化で減らした後に極大値を与える時刻を検出する．

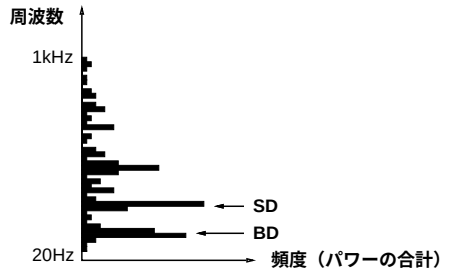

図 3: ヒストグラム中の BD と SD の特徴周波数

BD または SD がその時刻に発音したと判断する．BD と SD の発音時刻の信頼度は，その立ち上がり成分の $d(t, f)$ の，最近得られた最大のピーク値に対する相対値とする．

3.2 ビート予測

周波数解析の結果を複数のエージェントが解釈し，次のビート時刻をそれぞれが予測する．この解釈の結果は，次の解釈の統合処理へと集められ，最も確信度の高い解釈が選択される．各エージェントが出力する解釈の結果は，次のビート時刻，そのビートタイプ，ビートの間隔の三つから構成される (図 4)．

ビート予測のエージェント (その数を NumAGENTS 個とする) は，二つずつの組に分けられる．同じ組の二つのエージェントは協調し合いながら，両者が同じビートの間隔で，お互いにビートの間隔の 1/2 ずれた時刻を予測する．これにより，一方が 4 分音符の裏拍を予測していても他方が正しい表拍を予測できる．

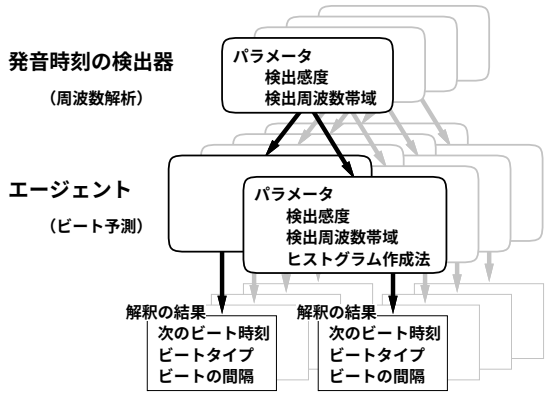

図 4: エージェントと発音時刻の検出器

各エージェントの組は，**3.1.3** で述べたように，それぞれ対応する検出器から発音時刻を受けとる (図 4)．エージェントの組は，自己評価した確信度が長期間低い場合に，対応する検出器の二つのパラメータを調整することができる．つまり，ビート予測のエージェント (上位) から発音時刻の検出器 (下位) へのフィードバックの機構を持つ．

それぞれのエージェントは，ビートを解釈する際の戦略を変更する三つのパラメータ (検出感度，検出周波数帯域，ヒストグラム作成法) を持つ．パラメータの値の設定は，各エージェントの組ごとに異なり，同じ組の中の二つのエージェントだけが同じ値をとる．こうして入力を各エージェントが異なる視点から調べることにより，様々なビートの解釈の結果を得ることができる．

1 番目と 2 番目のパラメータ (検出感度，検出周波数帯域) は，発音時刻の検出器内の対応するパラメータを制御し，エージェントが受けとるビートの手がかりの質を調整する．3 番目のパラメータ (ヒストグラム作成法) は，successive か alternate という値をとり，発音時刻の間隔 (IOI: inter-onset-interval) のヒストグラム (図 6) を作成する方法を変更する．successive の場合には，今回と前回の発音時刻の間隔の場所 (ヒストグラムの横軸) にそれらの信頼度の積を加え，alternate の場合には，今回と二つ前の発音時刻の間隔の場所にそれらの信頼度の積を加える．あるエージェントの確信度が長期間低い値のままならば，最も確信度の高いエージェントのパラメータに近付くように，これらのパラメータの値を調整する．

以下の節では，実際の解釈の方法と解釈の結果の統合について述べる．まず，各エージェントはビートの間隔を求めて次のビート時刻を予測し，解釈の確信度を自己評価する (**3.2.1**)．次に，予測したビートに対してビートタイプを判断し，解釈の確信度を補正する (**3.2.2**)．最後に，すべてのエージェントの解釈の結果の中で，最も確信度の高いものを選択する．(**3.2.3**)．

3.2.1 次のビート時刻の予測

各エージェントは，前回のビート時刻にビートの間隔を加えて次のビート時刻を予測する (図 5)．ビートの間隔は，高い信頼度を持った発音時刻の間隔の中で，最も良く出てくる間隔とする．実際には，IOI(発音時刻の間隔) のヒストグラムの最大ピークから求める (図 6)．もし前回のビート時刻とほぼ一致する発音時刻がある場合には，その発音時刻にビートの間隔を加える．

各エージェントは，解釈の確信度を自己評価する．

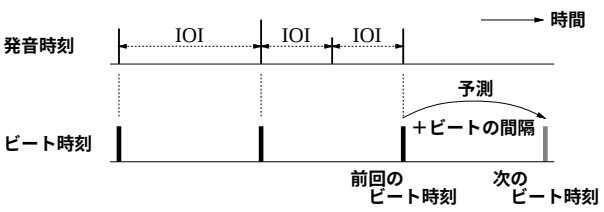

図 5: 次のビート時刻の予測

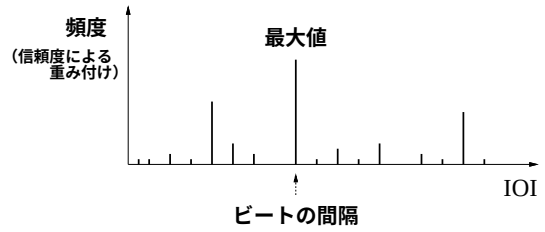

図 6: 発音時刻の間隔のヒストグラム

過去に予測したビート時刻に発音時刻が一致していれば，確信度を上げる．また，過去に予測したビート時刻の間の 8 分音符や 16 分音符に相当する時刻に，発音時刻が一致している場合にも，確信度を少し上げる．これらに一致しない場合には，確信度を下げる．

3.2.2 ビートタイプの判断

予測した各ビート時刻に対し，それぞれのビートタイプを判断する．本来は様々な手がかりに基づいて判断するのが望ましいが，現在の実装では，BD と SD の発音時刻だけからビートタイプを判断する．

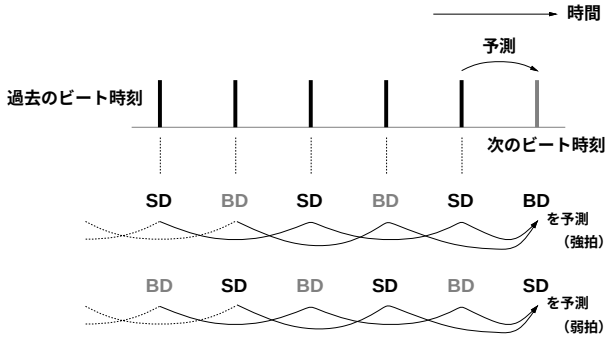

図 7: ビートタイプの判断

個々の BD と SD の検出は，ノイズや他の音によって妨害される可能性が高い．また，シンコペーションなどの場合に，BD や SD の音が仮定された場所で鳴らないことがある．そこで，BD と SD がある程度交互に鳴るという性質を用いて，次のビートに BD と SD のどちらが予測されるかを判断する．例えば，過去のビートの位置で BD SD ？ SD BD ？ ？ SD BD SD のように検出された場合には，次に BD が来ると予測する．この判断をおこなう際に，SD には BD よりも大きく重み付けしておく．これは，シンコペーションなどで BD がしばしば強拍からずれて鳴るのに対し，SD は弱拍で仮定通りに鳴る確率が高いからである．

もし次のビートが BD だと予測すれば，そのビートタイプを強拍とし，SD の場合には弱拍とする．そして，ビート時刻に BD と SD がある程度交互に検出されれば，その解釈の確信度を上げ，4 分音符に相当するビートの間隔を持つ解釈が選択されるようにする．

3.2.3 解釈の統合

すべてのエージェントから集められた解釈の結果は，ビート時刻とビートの間隔が同じもの同士グループ分けされる．各グループにおいて，グループ内の全解釈の確信度を合計し，そのグループの確信度とする．そして，最も確信度の高いグループ内の最も確信度の高い解釈を選択する．この結果は，ビート情報の生成へと送られ，そこで後処理とネットワークへ送出する準備がなされる．

4 実装

計算量が多い処理をリアルタイムにおこなうために，富士通の分散メモリ型並列計算機 AP1000 上に BTS を実装した (図 8)．AP1000 は，64 台のセルと呼ばれる要素プロセッサが，2 次元トーラス状に接続されている MIMD 型並列計算機である．BTS ではこれらのセルを 8 つのグループに分け，各グループに質的に異なる処理を直接割り当てる．各処理に割り当てられたセルの数を，図 8 の各長方形の右下に示す．矢印はグループ間の大局的なデータの流れを表しており，主に入力から出力へと U 字型にデータが流れる．

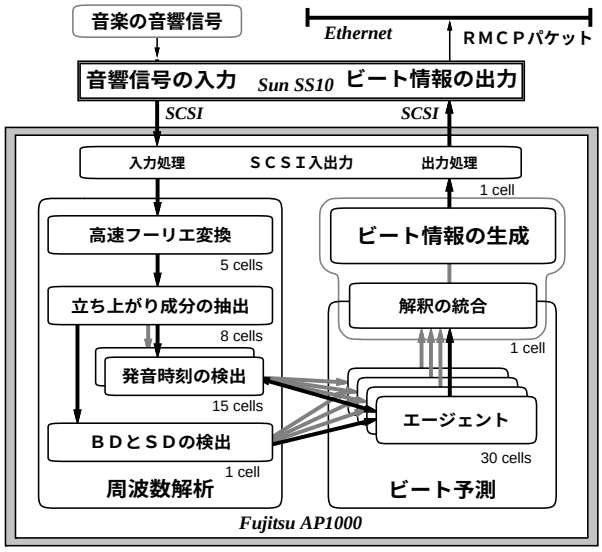

図 8: AP1000 への実装

現在の実装では，NumAGENTS (エージェントの数) は 30 個とした．したがって，発音時刻の検出器とエージェントの組の数は共に 15 個となる．これらのエージェントの組のパラメータの初期値を表 1 に示す．検出感度の単位は，次節で述べるフレーム時間である．

表 1: エージェントの組のパラメータの初期値

組 No.	検出感度	検出周波数帯域	ヒストグラム作成法
1	11	0-11kHz	successive
2	13	0-11kHz	successive
3	15	0-11kHz	successive
4	17	0-11kHz	successive
5	19	0-11kHz	successive
6	21	0-11kHz	successive
7	23	0-11kHz	successive
8	25	0-11kHz	successive
9	13	0-11kHz	alternate
10	19	0-11kHz	alternate
11	15	0-430Hz	successive
12	15	430-1.3kHz	successive
13	15	1.3k-3.0kHz	successive
14	15	3.0k-6.5kHz	successive
15	15	6.5k-11kHz	successive

4.1 音響信号の入力

音響信号の入力は，AP1000 に SCSI で接続されたワークステーション (Sun SS10) によりおこなう．音響信号を 22.05kHz, 16bit で A/D 変換し，256 点ごとにブロック化して BTS の全処理の基本単位とする．この 1 ブロックに相当する時間を 1 フレーム時間 (11.61msec) と呼び，フレーム時間を AP1000 内のすべての処理の時間単位とする．

現在の実装では，WinSIZE (FFT の観測区間) は 4 フレーム時間 (1024 点) とし，ShiSIZE (観測区間のずらし幅) は 1 フレーム時間 (256 点) とした．

4.2 ビート情報の出力, RMCP システム

ビート情報は，RMCP(Remote Music Control Protocol) パケットとして，予測されたビート時刻に Ethernet 上へ送信される．RMCP は，MIDI と LAN を融合した分散協調システム (RMCP システム) におけるサーバ・クライアント間の通信プロトコルである [10]．RMCP パケットとして送信 (ブロードキャスト) することで，ビート情報をネットワーク上の様々な機器で同時に活用できる．例えば，ワークステーションに接続された MIDI 楽器により，音楽に合わせてドラム音や手拍子の音を出力したり，グラフィックスワークステーションにより，音楽に同期した CG(Computer Graphics) を生成できる．

現在，RMCP システム上で，BTS の出力結果を視覚と聴覚の両方で確認できる (図 9)．Beat Information Display Server (BIDS) という RMCP サーバは，ビート情報を視覚化する．音楽に合わせてビート時刻とビートタイプを確認できるように，Ethernet を通じて受けたビート情報をカラー表示する．また，Beat Information Sound Server (BISS) という RMCP サーバは，ビート情報を聴覚化する．BISS は，MIDI 楽器を制御してドラム音や手拍子の音を出力する．ドラム音の場合には，強拍でバスドラム，弱拍でスネアドラムの音を鳴らす．もし，BTS が 16 分音符に相当するビート情報を送信した場合[3]には，8 分音符か 16 分音符の場所でハイハットシンバルの音を鳴らす．

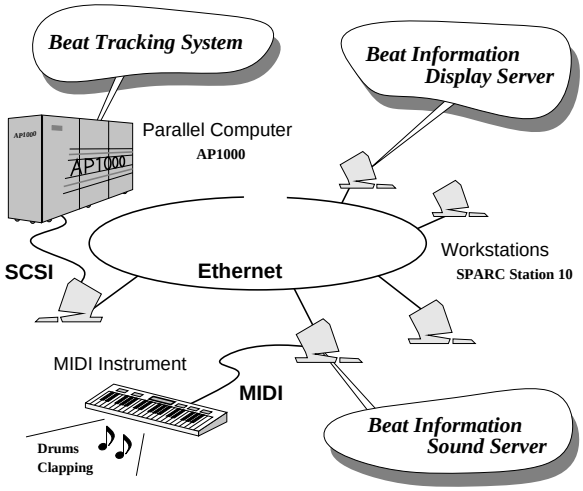

図 9: RMCP システム上の BTS，BIDS，BISS

5 実験結果

市販の CD からサンプリングしたモノラルの音響信号を用いて実験した．BD が強拍 (1，3 拍目)，SD が弱拍 (2，4 拍目) で多く鳴るロック・ポップスの，最初の 1〜2 分を入力した．実験は，テンポが 78〜168 M.M. の範囲でほぼ一定な 30 曲を対象におこなった．

表 2: 実験結果*

No.	曲名 (アーティスト)	結果	テンポ
1	ジュリアン (Princess Princess)	○	78
2	Dancing queen (ABBA)	○	99
3	もう恋なんてしない (槇原 敬之)	○	105
4	Something about you (LEVEL 42)	○	106
5	Pride (U2)	×	106
6	Need you tonight (INXS)	○	109
7	Satisfied (Richard Marx)	○	109
8	ラブストーリーは突然に (小田和正)	×	114
9	Open your heart (Madonna)	○	115
10	Black or white (Michael Jackson)	○	115
11	Sussudio (Phil Collins)	○	121
12	All that jazz (Breathe)	○	123
13	Raspberry Dream (Rebecca)	○	129
14	Any way you want it (Journey)	○	139
15	Seven years after (Princess Princess)	○	143
16	Be good to yourself (Journey)	○	151
17	Arcadia (T-Square)	×	156
18	Danger zone (Kenny Loggins)	○	157
19	Mighty wings (Cheap Trick)	○	159
20	Surfing with the alien (Joe Satriani)	○	168

* 表で示した以外にも，10 曲において正しいビートを得ることができた．

[3] BTS は，4 分音符のビート時刻から，外挿によって 16 分音符に相当するビート情報を出力する機能も持つ．これは，より短い時間間隔の同期が必要な場合に有効である．

実験結果を表 2 に示す．30 曲に対して実験した結果，27 曲に対して正しいビートを得ることができた．どの場合も曲の先頭では，BD と SD の特徴周波数を獲得していないために，ビート時刻を認識できてもビートタイプは判定できない．しかし，BD と SD がほぼ定常的に鳴り出して数小節経つと，ビートタイプも正しく出力するようになった．

5 番，8 番，17 番の曲において，ビートが正しく得られなかった原因を述べる．5 番の曲では， ビート時刻は正しく得られたが，ビートタイプをしばしば誤っていた．これは，BD に相当する低域のピークのパワーが低過ぎるために，BD の特徴周波数を正しく獲得できなかったからである．ヒストグラム中の本来 SD に相当するはずのピークが，最も低いピークとして検出されたため，本来の SD を BD とみなしていた．8 番の曲では， ビート時刻は正しく得られたが，ビートタイプで強拍と弱拍を逆に判断していた．ヒストグラム中の，最低周波数のピークとその上の最大パワーのピークの両者が，実際には BD と SD の両者に共通の周波数成分であったため，BD と SD の特徴周波数を正しく獲得できなかった．17 番の曲では， 曲の大半で正しいビートを認識できたが，途中で約 3 小節間ビートタイプを逆に判断していた．これは，ドラムスに変則的なリズムが一時期あったのが原因であった．

6 おわりに

本稿では，ロック・ポップスの音楽音響信号に対し，リアルタイムにビートを認識・予測するビートトラッキングシステム (BTS) について述べた．BTS は，ドラムスを含む複数の楽器で演奏された音響信号を扱うことができ，4 分音符に相当するビート情報を入力音楽に合わせて出力する．並列計算機 AP1000 上に実装して実験した結果，ドラム音の出現位置の制約と 4/4 拍子でテンポがほぼ一定という制約はあるが，30 曲中 27 曲に対して正しくビートトラッキングできることを確認した．

BTS ではビートの様々な解釈の可能性を同時に調べるために，複数のエージェントがそれぞれ異なった戦略で並列にビートを予測する．これにより，あるエージェントの解釈がはずれても，ビートを見失わずに正しい結果を出力できる．また，バスドラムとスネアドラムの特徴周波数を自動的に獲得して検出することで，各ビートが強拍か弱拍かを判断できる．

ビートトラッキングの技術は多くのマルチメディアアプリケーションに応用できる．例えば，ビデオ編集システムでは，映像トラックを音楽トラックに容易に同期させることができる．ハードディスクレコーディングシステムでは，自動インデキシング機能によりビート単位での編集が可能になる．さらに，音楽に同期した CG をリアルタイムに生成したり，人間の演奏に同期して照明などを制御することなどにも応用できる．

今後の研究としては，入力に対する制約を減らすために，ビート予測のアルゴリズムの改良を考えている．また，ドラム音の検出手法の改良や小節の先頭の認識などのより大局的な音楽構造の理解，実際のマルチメディアシステムへの応用などもおこなう予定である．

謝辞

AP1000 の音響入出力の実装について協力して頂いた富士通研究所の稲野 聡 氏に深く感謝いたします．また，AP1000 の実行環境を提供して頂いた富士通研究所 並列処理研究センターに感謝いたします．

参考文献

[1] Roger B. Dannenberg and Bernard Mont-Reynaud: *Following an Improvisation in Real Time*, Proceedings of the 1987 International Computer Music Conference, pp.241-248 (1987).

[2] Peter Desain and Henkjan Honing: *The Quantization of Musical Time: A Connectionist Approach*, Computer Music Journal, Vol.13, No.3, pp.56-66 (1989).

[3] Paul E. Allen and Roger B. Dannenberg: *Tracking Musical Beats in Real Time*, Proceedings of the 1990 International Computer Music Conference, pp.140-143 (1990).

[4] Anthonie Driesse: *Real-Time Tempo Tracking Using Rules to Analyze Rhythmic Qualities*, Proceedings of the 1991 International Computer Music Conference, pp.578-581 (1991).

[5] David Rosenthal: *Emulation of Human Rhythm Perception*, Computer Music Journal, Vol.16, No.1, pp.64-76 (1992).

[6] David Rosenthal: *Intelligent Rhythm Tracking*, Proceedings of the 1992 International Computer Music Conference, pp.227-230 (1992).

[7] David Rosenthal: *Machine Rhythm: Computer Emulation of Human Rhythm Perception*, Ph.D. Thesis, Massachusetts Institute of Technology (1992).

[8] W. Andrew Schloss: *On The Automatic Transcription of Percussive Music — From Acoustic Signal to High-Level Analysis*, Ph.D. Thesis, CCRMA, Stanford University (1985).

[9] 片寄 晴弘: 音楽感性情報処理に関する研究 , 大阪大学基礎工学部 博士論文, pp.45-48 (1991).

[10] 後藤 真孝, 橋本 裕司: MIDI 制御のための分散協調システム — 遠隔地間の合奏を目指して— , 情処研報, Vol.93, No.109, 音楽情報科学 93-MUS-4-1 (1993).