音 楽 情 報 科 学 16–6
（1996．7．28）

ハービー君: 演繹オブジェクト指向に基づいてジャズらしいコードにリハーモナイズするシステム

後藤 真孝
早稲田大学 理工学部
goto@muraoka.info.waseda.ac.jp

平田 圭二
NTT 基礎研究所
hirata@nefertiti.brl.ntt.jp

あらまし　本稿では，入力された単調なコード進行をジャズらしいコード進行にリハーモナイズするシステム「ハービー君」の設計方針，内部構成，実装について報告する．本研究では，演繹オブジェクト指向 (DOO) の枠組と音楽知識処理との親和性が高いことを実証し，DOO に基づく音楽応用システム構築法を確立するために，DOO に基づくジャズピアノ知識ベースシステムを応用してハービー君を構築する．リハーモナイズ前後のコード進行の関係を DOO におけるオブジェクト間の包摂関係で合理的かつ自然に表現することで，ハービー君はリハーモナイズ後のコード進行を推論できる．我々は，拡張性の高い汎用的なシステムであることを考慮しながら，ネットワーク上の分散システムとしてハービー君を実現した．

Herbie-kun: A Jazz Chord Reharmonizer in a Deductive Object-oriented Framework

Masataka Goto
School of Science and Engineering
Waseda University

Keiji Hirata
NTT Basic Research Laboratories

Abstract　This paper presents the design principles, the system structure and the implementation of a reharmonization system, *Herbie-kun*, which accepts a simple chord progression and generates a jazzy chord progression. The purpose of this research is to exhibit high affinity of a deductive object-oriented (DOO) framework with music knowledge processing, and to establish an implementation methodology of music application systems in the DOO framework. We hence develop Herbie-kun as an application system of a jazz piano knowledge base system employing the DOO framework. Herbie-kun can infer a reharmonized chord with using a subsumption relation of the DOO framework that rationally and naturally associates original and reharmonized chord progressions. We have implemented Herbie-kun as a distributed system on a LAN, keeping high expandability and availability.

1　はじめに

様々な知識表現手法を用いて，これまで多くの音楽知識処理システムが構築されて来た．音楽知識を表現し操作する手法・枠組として何を選択するかというのは，システムを実現する際の大切な design decision であるにも関わらず，それほど注意が払われてこなかった [1]．我々は，そのための手法・枠組として演繹オブジェクト指向 (以下，DOO: Deductive Object-Orientation) を取り上げ，実証システムとして DOO に基づくジャズピアノ知識ベースシステム (以下，ジャズピアノ KBS: Knowledge Base System) を構築してきた．本研究は，ジャズピアノ KBS およびその応用システムの構築を通じて，1) DOO の枠組と音楽知識処理との親和性が高いことを実証し，2) DOO に基づく音楽応用システム構築法を確立することを目的とする．

ジャズピアノ知識ベースとは，ジャズピアノに関する様々な音楽知識を蓄えており，外部からの問い合わせに柔軟に答えることのできるデータベースである．ジャズピアノ KBS を核にして，例えば，メロディに対するコード付け，ジャズらしいコード進行への変換 (リハーモナイズ)，コードネームから実際のボイシングへの変換，ソロピアノ譜への編曲などをおこなう音楽応用システムを構築することができる．さらに，本システムの将来構想としては，GUI (Graphical User Interface) による対話的な問い合わせと知識ベースの容易な更新管理，Internet などのネットワークを介し

た複数の知識ベースの協調問題解決，並列処理による高速化，実時間処理の達成などを目指している．

従来構築してきたジャズピアノ KBS [2, 3] を改良してこの将来構想を実現していく際に，その途中段階においても実用性のある，効果的な応用システムがあることが望ましい．そこで我々は，その第一段階となる基本的な問い合わせ用 GUI を従来のシステムに付与し拡張することで，応用システム「ハービー君」を試作する．その際，拡張性が高く実装が容易なようにシステムを設計することで，ハービー君をベースにシステムの将来構想が実現しやすくなるよう考慮する．

ハービー君は，入力された原曲のコード進行をジャズらしいコード進行にリハーモナイズするシステムである．リハーモナイズとは，あるコード進行と似た機能を持つ別のコード進行を求める操作で，一般にテンションを含んだより豊かな響きを持つコードに変える．ハービー君は，DOO に基づくジャズピアノ KBS を利用してこのリハーモナイズを実現する．リハーモナイズ前後のコード進行の関係を，DOO におけるオブジェクト間の包摂関係で合理的かつ自然に表現することで，リハーモナイズ後のコード進行を推論することができる．

ハービー君は，ユーザが指定した原曲のコード進行とリハーモナイズ後のコード進行を，画面表示と音楽演奏の両者によりユーザに提示する．リハーモナイズ前後のコードネームを上下に並べて表示することで，ユーザは結果を容易に対比できる．またコードを実際に演奏することで，テンションの付いたコードネームだけでは理解できないユーザでも，リハーモナイズ後の結果をコードの響きで確認することができる．

ハービー君の試作を通じて，DOO の枠組が，ユーザの意図している音楽的な概念・実体・関係を合理的かつ自然に記述するのに十分な表現力を提供できることが検証できた．また，ネットワーク上に分散した複数の計算機に，モジュラリティ高く機能単位ごとに異なるプロセスとして実装したことで，負荷分散の効果だけでなく高い拡張性，再利用性も得ることができた．

2 演繹オブジェクト指向に基づくジャズピアノ知識ベースシステム

我々はこれまで，DOO [4] に基づく音楽知識表現について検討を重ねてきた [2, 3]．DOO の枠組を用いると，様々な音楽的な概念や実体が合理的かつ自然に表現され，見通し良く操作することができる．DOO を用いれば，**1** で述べたようなジャズピアノ KBS を核にした様々な音楽応用システムを実現しようとするときに，その知識ベース部分を共有したまま，問い合わせなどをおこなうインタフェースを新たに付加するだけでよい．これにより，各々の音楽応用システムが持つ知識を有機的に統合することが可能となる．

DOO には様々な特長 (オブジェクトによる柔軟な知識表現，包摂関係などを用いたインスタンスに基づく検索，知識や問い合わせの同一の枠組での記述，モジュラリティを持った知識の記述) があるが，本章ではハービー君に関連して，特にオブジェクトによる柔軟な知識表現と包摂関係について概説する．

2.1 オブジェクトによる柔軟な知識表現

DOO の枠組では，音楽的な概念や実体 (音符やコードなど) はそれぞれがオブジェクトとして表現される．具体的な例を挙げながら，どのように柔軟に表現できるのかを述べていく．

まず，音符をオブジェクトとして表現する場合を考える．例えばピッチ C の音符を表現するときに，特定の鍵盤を意識しない抽象的な C と，5 番目のオクターブの特定の鍵盤である具体的な C とを区別したい．DOO では，前者を note(pitch = C)，後者を note(pitch = C, octave = 5) と記述できる．ここで，pitch や octave の部分を**属性名**と呼び，C や 5 の部分を**属性値**と呼ぶ．DOO では，このように記述したい内容に応じて属性名–属性値のペアを自由に増減できるため，制約が少なく表現力が高い．これらはオブジェクトを識別する際にも用いられ，**固有属性**と呼ばれる．

次に，コードネームをオブジェクトとして表現する場合を考える．Cm や $\mathrm{Cm7^{11}}$ というコードネームは，それぞれ chord_name(root = C, name = m)，chord_name(root = C, name = m7, tension = {11}) と記述できる．ここでさらに，ボイシング (コード構成音) も追加して記述する．ボイシングはコード自体を識別するのには影響を与えない付加的な情報と考えられるため，root や name とは併記せず，chord_name(root = C, name = m) / (notes = {C, E♭, G})，chord_name(root = C, name = m7, tension = 11) / (notes = {C, E♭, G, B♭, F}) のように / の右側に記述する．これらはオブジェクトの識別には用いられずにオブジェクト間の関連だけを表し，**非固有属性**と呼ばれる．

2.2 オブジェクト間の包摂関係

二つのオブジェクト p, q について，p は q より具体的であるときに $\mathrm{p} \sqsubseteq \mathrm{q}$ と表記し，これを**包摂関係**と呼ぶ．直観的には，具体的 $\sqsubseteq$ 抽象的，複雑 $\sqsubseteq$ 単純，特殊 $\sqsubseteq$ 一般とみなすことができる．例えば，前述した音符やコード間には，note(pitch = C, octave = 5) $\sqsubseteq$ note(pitch = C) や chord_name(root = C, name = m7, tension = {11}) $\sqsubseteq$ chord_name(root = C, name = m) といった関係がある．つまり，音符はオクターブの位置が明示されればより具体的になり，コード

ネームはテンション等が付加されればより具体的になることを表現している．この包摂関係を用いれば，あるオブジェクトより具体的なオブジェクトなどを，簡潔に求めることができる．

一般に知識ベースを構築するときには，専門家が直接書き下したルールや，システムに提示されたインスタンスから帰納したルールを知識ベースに格納したりする．しかし，このような知識をルールで表現する手法では，不用意な抽象化や重要な情報の捨象により，意図したような推論結果が得られないことがある．そこで，インスタンスをそのまま知識ベースに格納しておけば，システムに与えられた情報を最大限に活用することができる．包摂関係は，このようなインスタンスに基づく検索のための強力なオペレータとして用いることができる．

3　ハービー君

ハービー君は，DOO に基づくジャズピアノ KBS を利用し，入力されたコード進行をリハーモナイズするシステムである．これまで試作したジャズピアノ KBS [3] では，包摂関係などから成る問い合わせが並列論理型言語 KL1 で記述されていた．しかも演繹結果は KL1 の内部表現としてそのまま出力されていた．そのため，システムの内部構造を理解し，新たな問い合わせをおこなうたびに KL1 でプログラムし直す必要があった．また知識ベース構築時には，ジャズのソロピアノを採譜した譜面とその演奏に対するコメント，原曲のコード進行を，すべて KL1 で記述して入力する必要があった．ハービー君では，知識ベース構築時の問題は扱わず，GUI 化の第一段階として問い合わせにおける問題点を解決する．

3.1　システム入出力

ハービー君を使用するために，ユーザはジャズ用にテンションが付加されていない原曲のコードネームとその調を入力する．具体的には，「小節番号, 拍位置: コードネーム」を各行とするテキストファイルで与える (図 1)．また演奏確認用に，コードネームに対応するメロディーを標準 MIDI ファイルの形式で与えることができる．現在の実装では，このメロディーはリハーモナイズの処理において全く考慮されない．

ハービー君は，リハーモナイズ後のコード進行を，画面表示と音楽演奏の両者によりユーザに提示する．画面表示では，結果を対比しやすくするために，リハーモナイズの前と後のコードネームを上下に並べて表示する (図 1)．また，処理の途中でも結果の得られた部分から随時表示していく．音楽演奏では，リハーモナイズ前後のコードを，メロディーも合わせてそれぞれ演奏する．これにより，テンションの付いたコードネームを理解できないユーザでも，リハーモナイズ

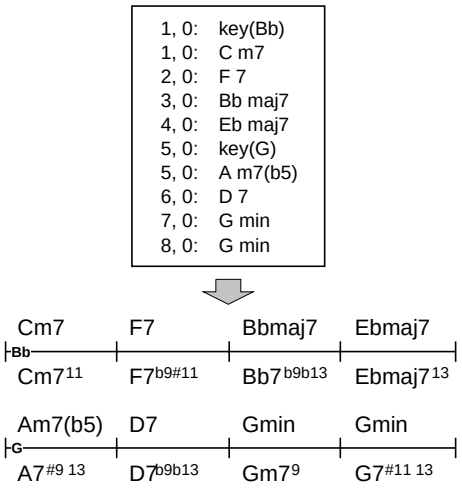

図 1: ハービー君の入出力

後の結果をコードの響きで確認することができる．その際，画面表示と同様に，処理の途中でも結果の得られた部分から随時演奏可能とする．

3.2　リハーモナイズの実現

あるコードをリハーモナイズするという処理を，DOO での演繹規則による推論で実現する．**2.2** で述べた包摂関係を用いれば「リハーモナイズされたコード ⊑ 原曲のコード」という対応付けができる．そこで，原曲のコードに対し，包摂関係に基づいてより具体化したコードをジャズピアノ KBS に問い合わせる操作により，リハーモナイズが実現できる．

ユーザから与えられた原曲のコードネームは，具体化の問い合わせとして，順次ジャズピアノ KBS に入力される．ジャズピアノ KBS は，基本的に **2** で述べたシステムの入出力部分を改良したものである．問い合わせに対し，同時に与えられた調を考慮しながら，知識ベース中にある具体化したコードネームを答えとして出力する．さらに，そのコードネームの非固有属性 (**2.1**) も検索することで，対応するボイシング (コード構成音) も合わせて出力する．このボイシングは音楽演奏の際に用いられる．

ジャズピアノ KBS には単独のコードネームに関する知識だけでなく，コード進行 (コードネームの系列) に関する知識も含まれている．そこで，短いコード進行から長いコード進行へと順に問い合わせていき，答の得られる最長一致したコード進行を演繹の結果とする．そして，その後の未決定のコード進行の先頭から次の演繹をおこなう．また，演繹されたコードネーム (コード進行) の候補が複数あるときには，現在の実装では乱数により一つを選択する．

3.3　処理構成

ハービー君の動作は，大きく分けて以下の七つの処理から成ると考えられる．

1. 問い合わせの発行
   原曲のコードネームを指定するテキストファイルに基づいて，ジャズピアノ KBS に具体化の問い合わせを順次発行する．
2. リハーモナイズ (ジャズピアノ KBS)
   ハービー君の核となる DOO に基づくジャズピアノ KBS である．
3. 問い合わせ／答えのコードネームを画面表示
   リハーモナイズ前後のコードネームを，上下に並べて順次表示する．
4. 問い合わせ／答えのコードを試聴
   リハーモナイズ前後のコードを，そのボイシングに基づいて演奏情報としてリアルタイムに出力する．その際，ユーザに与えられたメロディーも合わせて演奏する．
5. 演奏の開始／停止の指示
   リハーモナイズの前と後のそれぞれのコード進行に対して，現在コードが得られている範囲での演奏開始と終了とを指示する．
6. 演奏情報の聴覚化／視覚化
   4. の演奏処理が出力した演奏情報を，実際に MIDI 楽器を制御することで音として出力し，画面に表示した鍵盤の色を変えることで視覚化する．
7. 問い合わせのコードネームからボイシングに変換
   ユーザから与えられた原曲のコードネームに対し，演奏確認用のボイシングを付加する．これは演奏時におこなうのではなく，事前にテキストファイル上で変換・付加しておく．

4 分散環境での実装

1 で述べたシステムの将来構想を実現する際には，各処理やインタフェースの構成要素のモジュラリティを高め，それらがネットワーク上に分散できるように実装することが好ましい．このように分散した構成要素 (プロセス) がお互い通信して処理を進める方式は，以下のような利点を持つ．

- 高い拡張性
  計算量の多い問題を解く際，大きな計算能力を得るために必要に応じて複数の計算機を付加してシステムを拡張することができる．また，より多くの知識を利用するために，将来的に Internet 上の様々なジャズピアノ KBS を接続して大規模な分散データベースへと拡張していく際にも，このような実装は整合性が良い．
  さらに，小さな機能単位ごとに適切に分割して実装することで，既に実装されているプロセスに大きな変更を加えずに，新たな機能を容易に追加し拡張できる．例えば，ハービー君を拡張して将来構想を実現していく過程で，知識ベースの更新管理機能を追加する場合には，ジャズピアノ KBS を改良し更新管理を担当するプロセスを追加するだけで拡張できる．既に実現されている GUI や音楽演奏のプロセスには変更を加えなくてもよい．
- 容易な実装
  個々のプロセスは小さな機能だけ実現すればよいため実装しやすく，別のシステムを構築する際にも機能単位での再利用が容易になる．また，通信部分の規約を定めプロセス間の独立性を高めれば，円滑な共同開発ができる．その際，実現すべき機能に適したプログラミング言語を使いわけて実装してもよい．例えばハービー君の試作では，あるプログラマがジャズピアノ KBS を並列論理型言語 KL1 で記述している一方で，もう一人がその他の GUI 関連の各機能を C 言語で記述した．

現在の実装では，ハービー君はネットワーク (Ethernet) で接続された複数のワークステーション (SGI Indigo2, Indy など) 上で動作する．各処理を UNIX のプロセスとしてサーバ・クライアント・モデルに基づいて実装し，プロセス間の通信には次に述べる RMCP (Remote Music Control Protocol) [5] を用いた．

4.1 RMCP

RMCP は，MIDI と LAN を融合した分散協調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り，シンボル化された音楽情報をネットワークを通じて伝送するために設計された．本実装において RMCP は，次の三種類の情報を RMCP パケットとして伝送する役割をする．

- **コード情報** (Chord Information)
  ジャズピアノ KBS がコードを入出力するためのパケットで，問い合わせの種類／答えの種類 (現在は「具体化の問い合わせ」「具体化の答え」の二つのみ実装されている)，小節番号，拍位置，コードネーム，ボイシングにより構成される．曲の調の指定も伝送することができる．
- **演奏情報** (MIDI Information)
  演奏用の MIDI メッセージを含むパケットである．
- **演奏制御情報** (Control Information)
  演奏するプロセスに演奏の開始／停止を指示するための情報を含むパケットである．

クライアントが RMCP パケットをすべてのサーバへ向けて送信 (ブロードキャスト) することにより，問い合わせと答えのコード情報などをネットワーク上の様々なサーバで共有して同時に活用できる．

4.2 システム構成

RMCP により実装したシステムの構成図を図 2 に示す．**3.3** の処理の内，1. の問い合わせの発行を RMCP Chord Provider，2. のリハーモナイズを Jazz

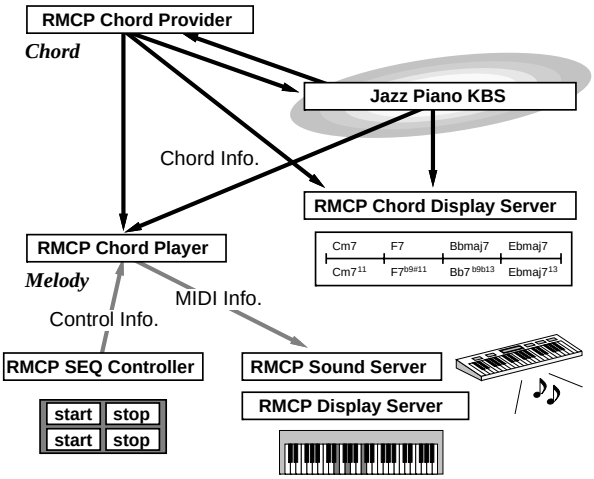

図 2: RMCP によるシステム構成図

Piano KBS[1]，3. のコードの画面表示を RMCP Chord Display Server，4. のコードの演奏を RMCP Chord Player，5. の演奏の開始／停止の指示を RMCP SEQ (Sequencer) Controller，6. の演奏情報の聴覚化／視覚化を RMCP Sound Server ／ RMCP Display Server がそれぞれ担当する．これらは，一つのジャズピアノ KBS のプロセスとその他の GUI 用プロセス群とみなせる．以下では，これらのプロセスについて順番に説明する．

4.2.1 RMCP Chord Provider

Jazz Piano KBS のプロセスに対し，原曲のコード情報を具体化の問い合わせとしてブロードキャストする．このコード情報は RMCP Chord Display Server にも同時に受けとられ画面表示される．現在の実装では，Jazz Piano KBS の出力 (答えのコード情報) に 8 小節分先行して問い合わせを発行する．

4.2.2 Jazz Piano KBS

原曲のコード情報を受信し，リハーモナイズ後のコードネームとそのボイシングを含むコード情報をブロードキャストする．現在の実装では知識ベースへの知識の格納・更新は RMCP を介しておこなうことはできず，事前に用意しておく必要がある．

4.2.3 RMCP Chord Display Server

RMCP Chord Provider から受信したリハーモナイズ前のコードネームと，Jazz Piano KBS から受信したリハーモナイズ後のコードネームを，各パケットが到着し次第順次表示する．これにより，問い合わせの発行状況と答えの出力状況を確認できる．

4.2.4 RMCP Chord Player / RMCP SEQ Controller

RMCP Chord Player は，受信したリハーモナイズ前後のコード情報を蓄積しておき，RMCP SEQ Controller からの演奏制御情報によって演奏を開始／停止する．演奏は，コード情報中のボイシングに基づいて MIDI メッセージを生成し，演奏情報としてブロードキャストすることでおこなう．その際，ユーザに事前に与えられたメロディーの標準 MIDI ファイルも演奏情報として合わせてブロードキャストする．

ユーザは，RMCP SEQ Controller を GUI として用いることで，リハーモナイズの前と後のそれぞれの演奏に対して開始と停止を指示できる．

4.2.5 RMCP Sound Server / RMCP Display Server

RMCP Sound Server は，演奏情報のパケット中の MIDI メッセージを MIDI 楽器に送ることにより，実際の演奏音として出力する．また，RMCP Display Server は，パケット中の MIDI メッセージに基づいて画面上の鍵盤の色を変えて，演奏状態を表示する．両者は文献 [5] の RMCP サーバをそのまま再利用している．

5 実験結果

現在我々が試作しているジャズピアノ KBS およびハービー君を運用して，実験をおこなった．まず実験のための準備として，KLIC [6] 上に実装されたジャズピアノ KBS に，Herbie Hancock が弾いたソロピアノを採譜した譜面とその演奏に対するコメント，原曲のコード進行 [7] を入力知識として与えた [3]．このソロピアノは，Autumn Leaves を 1 コーラス 32 小節演奏したものである．その結果合計約 380 個のオブジェクトが知識ベースに格納された．本システムには上記以外の背景知識 (機能和声理論など) は与えられていない．

ハービー君による実験結果の例を，図 3 と図 4 に示す．これらの図は，知識ベースに対する問い合わせと答えがすべて終了した時点での，リハーモナイズ前後のコードネームを RMCP Chord Display Server が表示した結果である．図 3 の Autumn Leaves は，知識ベース構築時と同様の曲である．一旦知識ベースに知識として格納されたものから演繹しているため，Herbie Hancock の演奏とは異なるリハーモナイズもなされている．図 4 の Summertime は，異なる曲に対してもリハーモナイズできることを確認するために実験した．

1 では研究目的として，1) DOO の枠組と音楽知識処理との親和性の高さを示す，2) DOO に基づく音楽応用システム構築法を確立する，の二つを挙げた．これらは単にシステムの出力するコード進行の音楽的善し悪しだけで評価は下せない．

1) の目的に関しては，リハーモナイズ前後のコード進行の関係を DOO の包摂関係を用いて簡潔に表現し，入出力のコード進行の比較から音楽的に意味のある計算をしていることを確認した．何を包摂関係で表現するかはユーザの恣意的解釈によるが，少なくとも

[1] 本稿では DOO に基づくジャズピアノ KBS を実装したプロセスを Jazz Piano KBS と表記して両者を区別する．

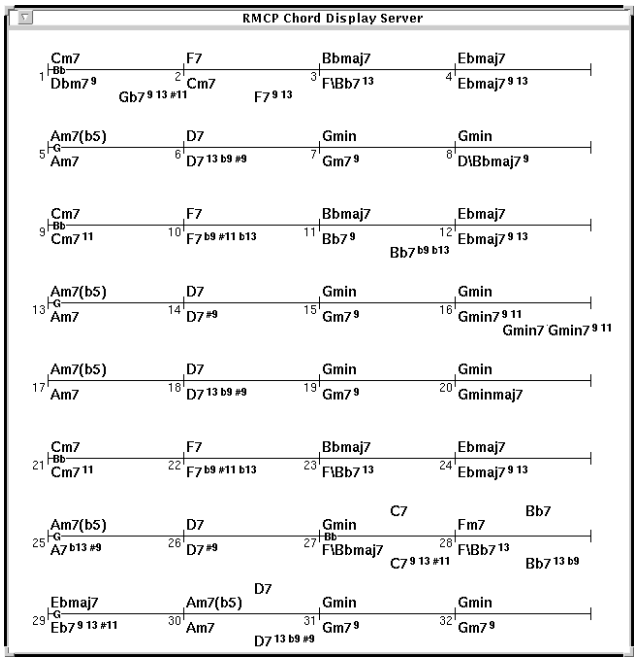

図 3: 実験結果 (Autumn Leaves: Joseph Kosma 作曲)

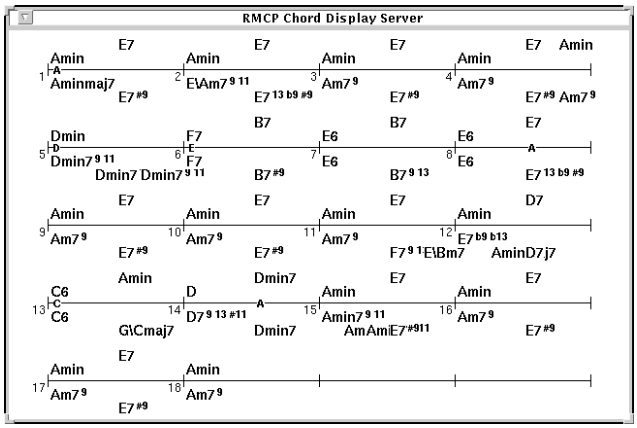

図 4: 実験結果 (Summertime: George Gershwin 作曲)

本実験の範囲では，ユーザの意図を合理的かつ自然に記述するのに DOO の枠組が十分な表現力を提供することができた．

2) の目的に関しては，各処理や GUI の構成要素をモジュラリティ高く機能単位ごとに異なるプロセスとして実装したことで，負荷分散の効果だけでなく高い拡張性，再利用性も得ることができた．現在 Jazz Piano KBS はネットワーク上に一つしか存在しないが，その問い合わせと答えを通信するプロトコルはできるだけ汎用に設計してあるので，今後のシステム拡張にも十分対応できると考えている．

6 おわりに

本稿では，演繹オブジェクト指向に基づくジャズピアノ知識ベースシステムに GUI を付加する際の一応用例として，ジャズらしいコードにリハーモナイズするシステム「ハービー君」について述べた．本研究では，具体化と非固有属性の問い合わせによってリハーモナイズを実現し，高い拡張性を考慮しながら分散環境で実装した．我々はこれにより，演繹オブジェクト指向が音楽知識処理に適していることを，応用の観点からも一つ支持できたと考えている．

本研究に関連する研究報告としては，コード進行が与えられるとジャズのピアノ用ボイシングを出力するシステムがある [8]．これは，音楽応用システムにおけるインタフェースのあり方，音楽知識の形式的な表現法を検討するために試作された．二つの動作モードで対話的に処理を進めるインタフェースが有効であること，論理型言語を用いて宣言的に知識が記述できたことが報告されている．コード進行が与えられるとランダムにメロディ，ベースライン，リズムを組み合わせるシステムも試作された [9]．ランダムな組み合わせという方法論でジャズの即興演奏が生成できるかどうかが検証されたが，答えは否定的であった．西洋音楽の一般的な認知モデルの構築と音楽における学習メカニズムの解明を目的として，メロディが音楽的に良いかを判定するシステムも試作された [10]．その結果，学習においては背景知識が重要であり，不完全なモデルだが出発点としては有望であることが示された．

ハービー君の今後の研究開発について述べる．改良点としては，前後のボイシングの流れの考慮，知識の効率的な入力法の開発などがある．さらに，**1** で述べたシステム全体の将来構想へ向けて，Jazz Piano KBS に対する問い合わせプロトコルの拡張，知識ベース更新管理の GUI 化をはじめとするシステム拡張をおこなう予定である．

謝 辞

3.3 で触れた問い合わせのコードネームからボイシングに変換する処理を実現する際に，冠野欣也氏のコードネーム解析プログラムを利用した．同氏に感謝する．本研究の機会を与えて下さった NTT 基礎研究所 石井健一郎氏に感謝する．

参 考 文 献

[1] R. Dannenberg. Music Representation Issues, Techniques, and Systems. *CMJ*, Vol. 17, No. 3, 1993.

[2] K. Hirata. Towards Formalizing Jazz Piano Knowledge with a Deductive Object-Oriented Approach. In *IJCAI'95 Workshop on AI and Music*, 1995.

[3] 平田圭二. 演繹オブジェクト指向に基づくジャズピアノ知識ベースシステムの試作. 情処研報 音楽情報科学 95-MUS-11-9, Vol. 95, No. 74, pp. 55–62, 1995.

[4] M. Kifer. Deductive and Object Data Languages: A Quest for Integration. In *Proc. of DOOD'95*, 1995.

[5] 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 音楽情報科学 93-MUS-4-1, Vol. 93, No. 109, pp. 1–8, 1993.

[6] 近山隆. *KLIC-2.001 User's Manual*. ICOT, 1995. (`http://www.icot.or.jp/`).

[7] Herbie Hancock. Private Piano Lessons. In *Jazz Life*, pp. 144–150. 立東社, 1989.

[8] K. Hirata and T. Aoyagi. How to Realize Jazz Feelings – A Logic Programming Approach –. In *Proc. of FGCS'88*. ICOT, 1988.

[9] P. N. Johnson-Laird. Jazz Improvization: A Theory at the Computational Level. In P. Howell et al., editor, *Representing Musical Structure*, pp. 291–325. Academic Press, 1991.

[10] G. Widmer. Qualitative perception modeling and intelligent musical learning. *CMJ*, Vol. 16, No. 2, 1992.