JAIST Repository
https://dspace.jaist.ac.jp/

Title	Tonal Pitch Spaceを用いた楽曲の和声解析
Author(s)	坂本, 鐘期
Citation	
Issue Date	2010-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/8931
Rights	
Description	Supervisor:東条敏, 情報科学研究科, 修士

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

修 士 論 文

Tonal Pitch Space を用いた楽曲の和声解析

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

坂本 鐘期

2010 年 3 月

修 士 論 文

Tonal Pitch Space を用いた楽曲の和声解析

指導教官　東条 敏 教授

審査委員主査　東条 敏 教授
審査委員　島津 明 教授
審査委員　白井 清昭 准教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

0710202 坂本 鐘期

提出年月: 2010 年 2 月

Copyright © 2010 by Shouki Sakamoto

概要

和声解析は音楽情報処理の一つの基礎である．例えば編曲を考えたとき，ただ無造作に和音を割り当てたのでは，そのほとんどが美しい進行とはならず聞き苦しいものとなる．これは音楽がある種の規則を持っていて，人間はその規則に従った和声進行を美しいと感じ，規則から外れた進行に違和感を覚えるためである．この和声進行に課せられる制約には，調・カデンツの規則といったものがあり，和音は前後関係に依存して成り立っているのである．

和声的要素の解析を行う手法は多く提案されているが，いずれも解析精度において改良の余地が残っている．また，従来の手法では，短い区間の転調を検知することができず，Generative Theory of Tonal Music (GTTM) などへの応用が難しかった．西田らは Head-driven Phrase Structure Grammar (HPSG) による構文解析を和音列に応用したルールベースの手法により，これらの問題を解決したが，カデンツの規則から逸脱した和声進行に対して解析が困難であるという欠点があった．

根源的な原因は和声そのものの難しさである．楽曲は一般的に和声学的な規則に従って作られているが，それは必ずしも絶対ではなく作曲者の意図により敢えて無視される場合があり得る．和声学とは，過去の楽曲から得られた良い進行についての情報をまとめたものであり，すなわち，経験則によって成り立っている．また，これは作曲者のために作られたものであり，楽曲を解析する用途での使用は当初想定されていなかった．しかしながら，音楽はその根本から曖昧なものなわけではない．一見すると曖昧で感性の領域にあるように見える音楽であるが，その根底には数学的構造が隠れている．

本研究では，和声解析の手法として Lerdahl の音楽理論 Tonal Pitch Space (TPS) を応用する．TPS は，調の五度圏や構成音の重なりといった音楽の数学的構造に立ち返り，慣習的な和声学の理論を再構成しようという試みである．TPS は和声進行の「良さ」を定量化することができるという，音楽解析において好ましい性質を備えており、その定量化はカデンツ規則に近似している．

本研究では，大きく分けて以下の二つのことを行った．

第一は，TPS についての整備である．TPS の主部分である和音間距離は，調の五度圏，和音の五度圏，そしてベーシックスペースという三つの要素より主に成り立っている．これらは既知の数学的な観点からの音楽要素かその応用である．また，和音間距離を補正するために調間距離が用いられる．調間距離は調の関係に基づいて作成された調性空間を参照している．これは近親調が近くに配置されたトーラス空間になっている．TPS は既知の音楽理論の組み合わせにより和音間・調間の定量化をおこなうという点で興味深い理論であるが，その実装例は少なく，特に調間距離・修正和音間距離まで実装した研究は無かった．本研究ではこれらの要素の実装をおこなうために TPS の各ルールについて，実装が容易な形へと原意をできるかぎり保持する形での修正をおこなった．

第二に，TPS についての実装を元に和声解析システムを提案し，その評価実験をおこなった．和声解析システムは楽曲のコードシーケンスから適合する和音を列挙し，それらから考えられる和声進行解釈全体を示す和音候補グラフを構築し，TPS をもちいて和音候補間の距離を定量化することにより最短経路をもとめ，それをコードシーケンスに対する和声進行解として出力する．最短経路の選択にはダイクストラ法を用い，グラフの出力には Graphviz を使用した．提案システムの評価を二種類の方法によりおこなった．第一に，和声学の教科書に掲載された代表的なカデンツ 19 種を元に生成されたコードシーケンスを入力として与え，各コードシーケンスの調性を正しく認識できるかを確認した．その結果，提案システムは 19 種中 18 パターンについて正しく調性を認識し，残りの 1 パターンのみについて誤答した．しかしながら，この 1 パターンについても複合カデンツ上に出現した場合には正しく認識できることが明らかになった．第二に，実際の楽曲を用いての実験をおこなった．実験には，ベートーベンのピアノ・ソナタ Op.49-1，Op.49-2 の合計 4 楽章を対象として用いた．実験の結果，提案システムは 83.6%〜93.1%の再現率を導き，平均すると 1 楽曲あたりの再現率は 88.4%となった．

評価実験の結果より，システムはほぼ全てのカデンツを認識することができるなど，和声学との一致を示し，実際の楽曲をもちいた和声解析でも応用に十分な値を示した．これにより TPS の有用性について示すことができたと考える．

目次

第1章　はじめに

和声解析は音楽情報処理の一つの基礎である．

例えば，三つの和音からなる短い楽譜 (例えば起立・礼・着席の伴奏) を考える．この楽譜の中央部分の和音を変更して新しい和声進行を作ろうとしたとき，ただ無造作に和音を割り当てたのでは，そのほとんどが美しい進行とはならず聞き苦しいものとなる．これは，音楽がある種の規則を持っていて，人間はその規則に従った和声進行を美しいと感じ，規則から外れた進行に違和感を覚えるためである．

この和声進行に課せられる制約の一つが調であり，楽曲の全ての部分はそれぞれ何らかの調に属している．調とは，半音階の中から相性の良い 7 つの音 (調の構成音) を抜き出してできた集合であり，原則的に和音列はそれら 7 つの音のみから作られる．また，和音はその基盤とする (根音となる) 調の構成音によって，それぞれの調において 7 種類に分類することができる．

では，先ほどの楽譜の全体がハ長調であることがわかったとき，ハ長調が持つ 7 種類の和音のどれもが問題の部分に挿入可能だろうか．これもまた否である．7 つの和音はさらに 3 つの機能 (トニック，ドミナント，サブドミナント) のどれかに分類でき，ある機能の次にどの機能が来れば美しい和声進行となるかは，カデンツの規則により定まっている．またさらに，ある機能でも特定の和音の場合には認められないといったようにカデンツにはより詳細な制約が存在する [19]．

このように和音は単独で建っているのではなく，前後との関係によって成り立っている．そして和声進行は聴き手の認知に重大な影響を与える楽曲の骨組みとも言える部分であり，和音の解析は音楽情報を計算機で扱うためには必須の部分であるといえる．

和声解析は上述のような場合に限らず，さまざまな分野への応用が可能である．楽曲の検索においては和声進行や調によって分類することが期待でき，編曲や作曲を支援するシステムとして楽曲の和声情報や推奨される進行を提示することが考えられる．また，リアルタイムのシステムでも，人間の演奏に対して伴奏を生成して付与するなど，自動演奏に生かすこともできる．

また，和声解析の技術を必要としている理論として，注目したいのが Fred Lerdahl と Ray Jackendoff が 1983 年に提唱した A Generative Theory of Tonal Music (GTTM)[10] である．GTTM は，楽譜からそれぞれのイベント (音の変化) の認知的重要性を考慮して，それらの間に主従関係を定めることで，楽曲を二分木化する (図 1.1)．この簡約という概念を用いることにより，さまざまアプリケーションへ応用できることが期待され，GTTM の木構造を利用し二つの楽曲をモーフィングするシステム [7] や実際の演奏に合わせてそ

の後の展開を予測し提示する即興演奏支援システム [6] が実際に開発されている．しかしながら，現在のところ GTTM の実装は完全とは言えない．GTTM の理論は複雑で人間の直感的な判断を必要とする部分があり，それらを自動化するのが困難だからである．現在，最も実装状況が進んでいるものは浜中・平田・東条らのグループが開発している Full Automatic Time-span Tree Analyzer (FATTA)[5] であるが，これは GTTM の四つのサブ理論 (グルーピング構造解析，拍節構造解析，タイムスパン還元，プロロンゲーション還元) のうち三つ目まで実装している．四つ目のサブ理論であるプロロンゲーション還元は，すなわちタイムスパン還元までで得られた二分木を楽曲の和声的構造を考慮して組み直す作業であり，この作業のためには和声についての詳細な解析が必須となる．そこで本研究では GTTM に組み込み可能な詳細な和声解析を行う手法を提案する．

和声解析についての研究は幾つかの観点から分類可能である．観点の一つは，研究対象として wav ファイルなどのサンプリングされた音声データをとるか，MusicXML[4] ファイルや Standard MIDI ファイルのような記号化された楽譜データを取るかである．前者の研究としては，確率論的モデルを採用したものが多い．Krumhansl[8] や Sheh ら [15] の研究がそうであり，楽曲中の各ピッチクラスの発音時間を頻度としてベクトル化し，学習データとの比較を行うもので，Hidden Markov Model (HMM) が採用されている場合が多い．後者の研究は，音声を扱ったものと同様に Paiement ら [12] や Rhodes ら [13] のように学習と確率的手法をもちいる研究と，それ以外のルールベースの方法とに，更に二分できる．本研究は，最後の部分，すなわち記号的楽譜データを扱いルールベースの手法を用いる研究に属する．

同カテゴリーに属するものとして，長嶋らは，旋律の第一音を主音と見なして調認識する，旋律の最終音を主音と見なして調認識する，出現頻度の最も高い音を主音と見なして調認識をするという，三つの古典的手法を紹介している [20]．Longuet-Higgins と Steedman は調の構成音をマトリックスで表現し，スケールアウトするかを逐次検証する方法での調認識を行った [11]．また，近年では Temperley らが動的計画法を用いた和声解析手法を提案している [16]．

このように和声的要素の解析を行う手法は多く提案されているが，いずれも解析精度において改良の余地が残っている．また，これら従来の手法では，短い区間の転調を検知することができず，GTTM への応用が難しかった．西田ら [18] は Head-driven Phrase Structure Grammar (HPSG) による構文解析を和音列に応用したルールベースの手法により，これらの問題を解決したが，カデンツの規則から逸脱した和声進行に対して解析が困難であるという欠点があった．

これら根源的な原因は和声そのものの難しさである．楽曲は一般的に和声学的な規則に従って作られているが，それは必ずしも絶対ではなく作曲者の意図により敢えて無視される場合があり得る．和声学とは，過去の楽曲から得られた良い進行についての情報をまとめたものであり，そなわち，経験則によって成り立っている．また，これは作曲者のために作られたものであり，楽曲を解析する用途での使用は当初想定されていなかった．曲に対して和声付けを人手で行おうとしたとき，和声学に習熟した者であっても必ずしも見解

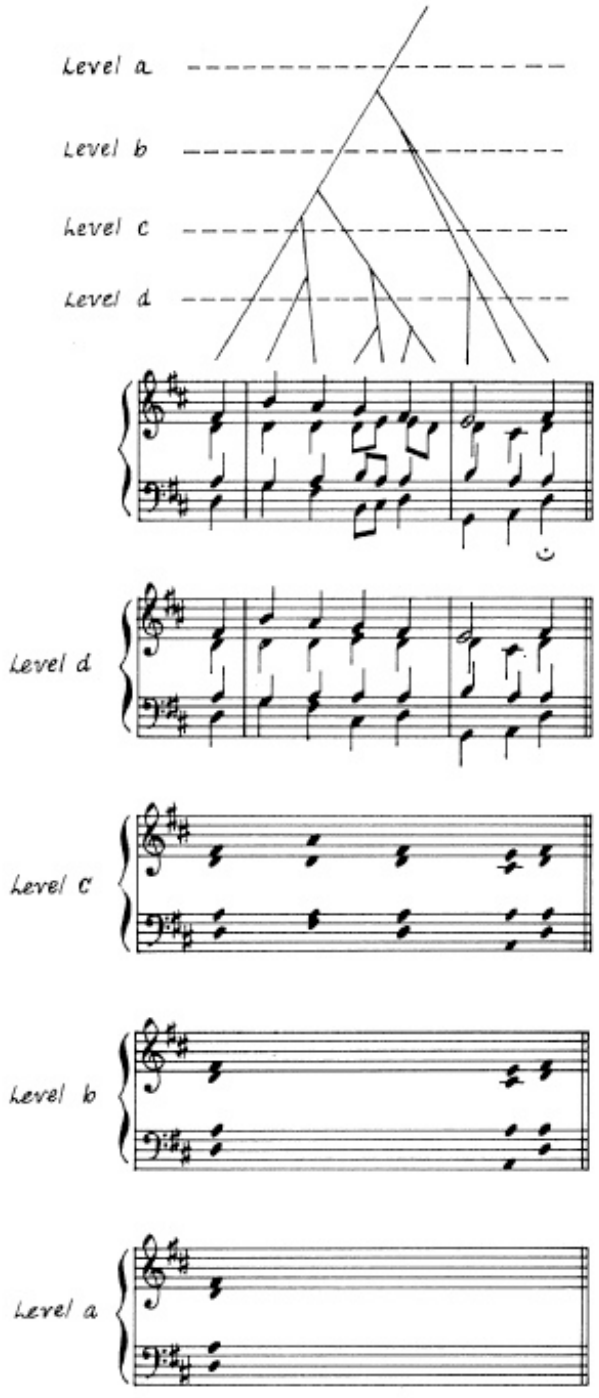

図 1.1: GTTM による簡約のイメージ ([9]pp.115)

が一致するというものではない．このように，和声とは曖昧なものであり，我々は直感的にはその裏にルールが存在することを理解しているものの，それは完全に明文化することはできずにいるのである．

しかしながら，音楽はその根本から曖昧なものなわけではない．「万物の根源は数である」と考えたピタゴラスは，二つの弦の長さ，すなわち周波数が簡単な整数の比であるときほどよく響くことを発見した．ここから作られたピタゴラス音律は今日使われている平均律の原型であり西洋音楽の基礎となっている．このように，一見すると曖昧で感性の領域にあるように見える音楽であるが，その根底には数学的構造が潜んでいる [17][2]．本研究では，和声解析の手法として Lerdahl の提唱する音楽理論 Tonal Pitch Space (TPS)[9] を応用する．TPS は，調の五度圏や構成音の重なりといった音楽の数学的構造に立ち返り，慣習的な和声学の理論を再構成しようという試みであると評価できる．カデンツ規則が西田らによって HPSG で表現されたように，ある進行を受容するかしないかというように 2 値的に和声進行を扱うのに対して，TPS はカデンツ規則に類似しながらも，その進行の「良さ」を定量化するという音楽解析に応用する上でより好ましい性質を備えている．そこで本研究では，この TPS による和音の距離の定量化を用いた楽曲から和声的要素 (調・機能・カデンツ構造) を解析するシステムを提案する．

本稿の構成は次のとおりである．第 2 章で本研究の核となる TPS における距離について説明した後，第 3 章で実装を主眼とした TPS の整備を行う．続いて，第 4 章で提案システムについて説明し，第 5 章で提案システムをもちいた実験について述べる．最後に，第 6 章で本研究についてまとめ，今後の課題について述べる．

第2章　Tonal Pitch Space

本章では，Lerdahl の Tonal Pitch Space における和音間距離，調間距離について述べる．まず，第 2.1 節で TPS について概要を述べたのち，第 2.2 節でピッチ・クラスをはじめとした関連する概念を解説し，第 3.5 節と第 2.4 節で，和音間距離と調間距離をそれぞれ定める．第 2.5 節では，第 3.5 節での和音間距離の定義では一般の音楽理論との乖離が生じることを示し，調間距離を用いてその修正を行う．最後に第 2.6 節では，TPS の特性についてより深く論じる．

2.1　TPS 概要

Lerdahl と Jackendoff の提案した GTTM は認知的な観点から楽曲の木構造を構築するルールベースの理論であり，実際にその計算機への部分的実装が行われ，自動編曲などの分野で活用されはじめている．しかしながら，GTTM のルールは複雑であり，また細部について音楽的で直観的な判断が必要となっており，その完全な実装は困難であった．このような問題点を解決するために，GTTM を基礎としてその補完を行うための認知科学的な音楽理論として Lerdahl により発表されたのが TPS である．

TPS は，ピッチ，和音，或いは調といった音楽要素の変化を聞き手がどのように捉えているかを量化して表現する認知科学的モデルである．これにより，どちらの音がある音からより近く (違和感なく) 感じるか，どちらの和声進行の方がより自然に感じられるかのように，従来の音楽理論では不可能であった音高，和音，調の比較が可能となる．また，これらの量化は，調の五度圏や和音の構成音のヒエラルキーなど既知の音楽理論を基礎とした簡単な数式により，多くの作曲における法則などを説明できる．

TPS のルールは多岐に渡るが，以下では本研究で用いる TPS の根幹部分である和音間距離と調間距離について主に述べる．なお，TPS の定義には数式的な曖昧性を含んだ部分があるが，本章ではとりあえず原著に従った形で述べる．それらの問題は，次章にて実装とあわせて解決を試みている．

2.2　ピッチクラスとベーシックスペース

楽譜中の音はそれぞれ周波数を持っており，その周波数はオクターブと音名により特定が可能である．また前述のようにオクターブの異なる同一音名の音は倍音により同一にみ

ることができる．基本的に，TPS による計算は音高のオクターブによる差異が生じないため，オクターブを無視して同一音名の音をクラス分けして考える．これをピッチ・クラスと呼ぶ．また，ピッチ・クラスには音名に対応して 0 から 11 の整数値を順番に割り振る (図 2.1)．例えば，オクターブ 4 の F とオクターブ 5 の F は，同じ F のクラスに属しており，そのピッチ・クラス値は 5 である．同様に，オクターブ 3 の C♯とオクターブ 5 の D ♭も同一のピッチ・クラスに属しており，ピッチ・クラス値は 1 となる．このように，C♯と D ♭のような異名同音も同じクラスに属しており，以下の計算についてもその差異は生じない[1]．

表 2.1: 音名とピッチクラス値

音名	C	C♯/D♭	D	D♯/E♭	E	F	F♯/G♭	G	G♯/A♭	A	A♯/B♭	B
値	0	1	2	3	4	5	6	7	8	9	10	11

また，和音の構成音のヒエラルキーを表現する手段としてベーシックスペースという概念を導入する．これは，ある音がその和音のなかでどの程度の影響力をもっているかということを示すものであり，例えば I/**C** の和音は図 2.2 のベーシックスペースにより表すことができる．ベーシックスペースは，例のようにピッチクラスを横軸，その影響度の level を縦軸とした二次元空間である．これは level ごとに次のようなことを示している．

- level a にあるピッチクラスはその和音の根音である．
- level b にあるピッチクラスはその和音の根音と五度音の二つである．
- level c にあるピッチクラスはその和音の構成音である．
- level d にあるピッチクラスはその和音が置かれている調の構成音である．

すなわち，ベーシックスペースで上にくる音ほど，和音を特徴づける影響力が高いと言える．ベーシックスペースのもう一つの特徴は和音の構成音だけでなく，その置かれている調もまた表現される点である．これらは，実際に発音されていないものの，その和音の印象づける上で，前後の関係として，暗黙的に響いている音であると考えることができる．

同様に，V/**C** と vi/**F** のベーシックスペースを図 2.2，図 2.2 に示す．

2.3 和音間距離

和音間の距離は複数の段階を経て定義される．まず最初に同一調の二つの和音の間の距離を求めることに限定して考え，ルール化を行う．これを**初版の和音間距離ルール**と呼ぶこととする．

[1]TPS においても，全ての場面において同音異名やオクターブの差異が無視されるわけではない．例えば，本稿では用いなかったが，音高間の距離の計算では異名同音が別々のオブジェクトとして解釈される．また，アトラクションの計算ではオクターブによる差異が生じる．

level a:	0							7			
level b:	0							7			
level c:	0				4			7			
level d:	0		2		4	5		7		9		11
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.1: ベーシックスペース I/**C**

level a:								7			
level b:			2					7			
level c:			2					7			11
level d:	0		2		4	5		7		9		11
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.2: ベーシックスペース vi/**F**

level a:			2								
level b:			2							9	
level c:			2			5				9	
level d:	0		2		4	5		7		9	10	
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.3: ベーシックスペース V/**C**

初版の和音間距離ルールは，和音の五度圏と共通の音という二つの似通った要素より計算される．これらは，心理音響学と郡論に基づいて採用されたものである．和音の五度圏は良く響く音が隣り合うように全音階を円状に並べたものであり，二つの和音がこの円状で近いほど良いと言える．また同様に両者のピッチクラスを比べたときに，共通の音が多いほどよいとされる．

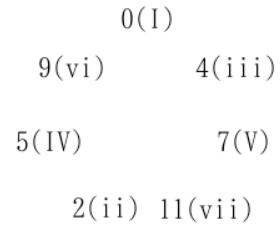

図 2.4: 和音の五度圏

和音の五度圏での数は，片方のベーシックスペースに対して次のオペレーションを何回行えば level a-c が重なるかによって計算される:

> **和音の五度圏ルール**　ベーシックスペースの level a-c にあるピッチクラスを level d の上で数えて右か左に (ベーシックスペースは左右にローテーションするものとして)4 マス動かす.

このオペレーションをあるベーシックスペースに一度適用すると，それは和音の五度圏で隣り合う和音のベーシックスペースと重なることとなる.

もう一つの要素である共通の音は，二つの和音が与えられたとき，両者のベーシックスペースを重ね合わせて異なっている部分の数である．例えば，ベーシックスペースではなく単純に音の違いで考えたとき，I と V では 1 箇所のピッチクラスのみ一致しているのに対して，I と vi では 2 箇所で一致している．しかしながら，各ピッチクラスの重要度を加味してベーシックスペースで比べると両者は等しくなる (図 2.3 と図 2.3 の強調部分).

level a:							*7*					
level b:			*2*				7					
level c:			*2*				7					*11*
level d:	0		2		4	5	7		9			11
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.5: ベーシックスペース I/**C** と V/**C** の比較

和音の五度圏とベーシックスペースの違いの数の二つを組み合わせたルールが以下のものである.

> **和音間距離ルール (初版)**　$\delta(x \to y) = j + k$, ここで $\delta(x \to y)$ は和音 x と和音 y の間の距離; j は x の和音部を y の和音部に重ねるのに必要な和音の五度

圏ルールの適用回数; k は x のベーシックスペースと y のベーシックスペースを比較したときに異なっているピッチクラスの数である.

幾つか例示する. まず上記のルールからわかるように, 同じ和音間の距離は 0 となる ($\delta(\mathrm{I} \to \mathrm{I}) = 0 + 0 = 0$). 先ほどの図 2.3 と図 2.3 は, それぞれ $\delta(\mathrm{I} \to \mathrm{V}) = 1 + 4 = 5$,$\delta(\mathrm{I} \to \mathrm{vi}) = 3 + 4 = 7$ となり, 構成音の違いの数では vi の方が近かったものが, ベーシックスペースと和音の五度圏による距離計算では V の方が近くなる. 和音間距離が vi よりも V の方がより近いというのは, 和声学的な認識と一致していると言える.

以上をもとに長調における I から同一調の三和音 I〜vii$^{\circ}$ の距離を計算すると表 2.2 のようになる. こちらも共に五度音である V と IV が同じ距離になるなど和声学との一致している.

表 2.2: 長調における和音間距離

和音	距離
I	0
ii	8
iii	7
IV	5
V	5
vi	7
vii$^{\circ}$	8

続いて, 初版の和音間距離ルールを拡張する形で, 調のことなる和音への対応を試みる. このためにまず, 調の違いを和音の五度圏のルールと同じように, 今度はベーシックスペースの全音階のレベル (level d) のシフトによって以下のようなオペレーションにより計算することとする.

> **調の五度圏ルール**　ベーシックスペースの level d にあるピッチクラスを level e の上で右か左に (ベーシックスペースは左右にローテーションするものとして) 動かす.

これは調の五度圏 (図 2.7) をベーシックスペース上のオペレーション化したものである. 調の五度圏で隣り合っている調同士は, 主音の周波数比が 2:3 になっており近親調 (属調・下属調) の関係にある. 主音のピッチクラス値で考えるならば, 片方のピッチクラス値を ±7 することにより, 属調・下属調のピッチクラス値となる.

この調の五度圏ルールを加味して初版の和音間距離ルールを改訂することで, 異なる調の和音間にも適用できる**完全版の和音間距離ルール**を次のように定める[2].

[2]次節で述べるように, 実際にはこのルールでは和声学との乖離が生じる. あくまで, 引数として全域的に和音をとることができるという意味においての「完全版」であると言える.

level a:										*9*		
level b:					*4*					*9*		
level c:	0				4					*9*		
level d:	0		2		4	5		7		9		11
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.6: ベーシックスペース I/**C** と vi/**C** の比較

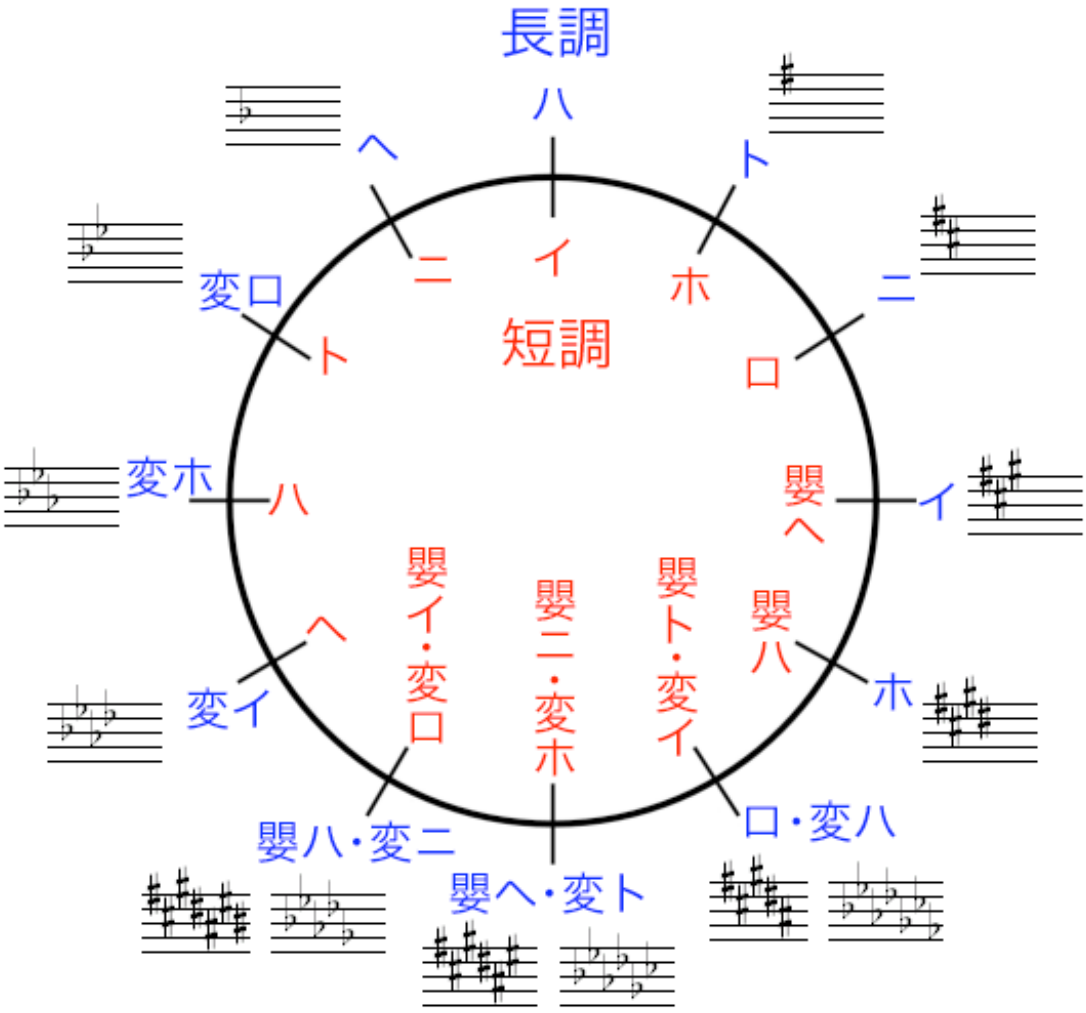

図 2.7: 調の五度圏 [1]

和音間距離ルール (完全版)　$\delta(x \to y) = i + j + k$, ここで $\delta(x \to y)$ は和音 x と和音 y の間の距離; i は x の音階を y の音階に重ねるのに必要な調の五度圏ルールの適用回数; j は x の和音部を y の和音部に重ねるのに必要な和音の五度圏ルールの適用回数; k は x のベーシックスペースと y のベーシックスペースを比較したときに異なっているピッチクラスの数である.

このルールは初版の和音間距離ルールを拡張したものである. 二つの和音が同じ調に置かれているときに距離は, 調の五度圏における距離 (i) が 0 であるため, 初版のルールと完全版のルールで同じ値になる.

完全版の和音間距離により, 転調のある和音間での距離を求めた例を幾つか示す. 図 2.3 はハ長調のトニック (I/**C**) とヘ長調のトニック (I/**F**) の比較である. 調の五度圏では 1, 和音の五度圏では 1, ベーシックスペースでは 5 だけ異なっており, 距離はそれらの合計で 7 となる. ベーシックスペースの比較をみると調の構成音のレベル (level d) で差異が生じていることがわかる. 図 2.3 のハ長調のトニック (I/**C**) とニ短調のトニック (i/**d**) の比較では, 調の五度圏で 1, 和音の五度圏で 2, ベーシックスペースでは 7, 異なっているので合計は 10 となる. 調 **F** と調 **d** は構成音が等しい並行調であるが, 主音の違いから距離には差が生まれている. I/**F** と i/**d** では, I/**F** の方が I/**C** からの距離が小さいため, より自然な進行であると言える.

level a:					*5*						
level b:	0				*5*						
level c:	0				*5*				*9*		
level d:	0		2		4	5		7		9	*10*
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.8: $\delta(\mathrm{I}/\mathbf{C} \to \mathrm{I}/\mathbf{IV}) = 1 + 1 + 5 = 7$

level a:			*2*								
level b:			*2*							*9*	
level c:			*2*			*5*				9	
level d:	0		2		4	5		7		9	*10*
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.9: $\delta(\mathrm{I}/\mathbf{C} \to \mathrm{i}/\mathbf{d}) = 1 + 2 + 7 = 10$

同様に, ト長調のトニック (I/**V**) への距離は, $\delta(\mathrm{I}/\mathbf{C} \to \mathrm{I}/\mathbf{V}) = 1 + 1 + 5 = 7$ となる (図 2.3). ここで, I/**V** は先述の V/**C**(図 2.3) とコードネーム (すなわち主音と構成音) が共に G(構成音ソシレ) で等しい和音であることに注目したい. 両者は発音される音としては同一であるが, その部分の置かれている調の違いにより距離に差が生じている. このように, TPS の距離は和音の調をどのように解釈するかによって異なる値となる. 一般に,

転調を含まないように解釈した場合 (この場合では $\delta(\mathrm{I}/\mathbf{C} \to \mathrm{V}/\mathbf{C})$) の方が距離は小さくなる．

最後に図 2.3 はト長調の第五和音への距離である．この例では，進行先の和音が三和音ではなく四和音 (セブンス・コード) となっている．このように TPS では一般的な三和音に限らず，セブンス・コードなどのより複雑な和音に対しても距離を計算できる．これらの和音の型の違いはベーシックスペースでの違いとして表現され，和音構成音のレベル (level c) に違いが生じる．それにより距離にも変動がおこる．

2.4 調間距離

上述の和音間距離は概ね和声学的な認識と近似した結果を示し，和音間距離の定量化という点では成功であった．しかしながら，和声学的な新式との乖離が生じる部分も存在する．それは特に二つの和音が遠隔調の関係にある場合である．例えば，$\delta(\mathrm{I}/\mathbf{C} \to \mathrm{III}/\mathbf{c}\#)$ は $4+1+7=12$ となるが (図 2.4)，実際にはいままで見てきた和声進行からかけ離れた和声学的に許されない進行である．このように完全版の和音間距離ルールでは二つの和音の調が離れるほど計算結果が不明瞭なものとなってしまう．そこで，和音間の距離に続いて，調間についても距離の量化を行い，それを用いて和音間距離のルールを修正する．

この為に，最初に**調性空間**(regional space) を構築する．調性空間はどの調とどの調が近い関係にあるのかを示すもので，これは各調のトニック (I の和音) と和音間距離のルールより求められる．これは次のような手順により求められる:

1. ある調を選択し，それを調性空間におく．
2. 選択された調のトニック (I/**I**) からの他のトニックへの和音間距離を計算したとき，その値が最小 (7) となるものが 4 つある (I/**V**, I/**IV**, I/**vi**, i/**i**)．これらは選択された調と近親調であり，属調・下属調・平行調・同主調の関係にある．これらを関係に応じて上下左右に配置する (属調ならば上，下属調は下，平行調・同主調は左右[3]; 図 2.13)．
3. 2 で新しく置かれた調を選択し，それらに対して 2 を行う．同様に全ての調が選択され置かれるまで繰り返す．

以上の手順により得られた調性空間が図 2.14 と図 2.15 である．これは，上下・左右がそれぞれ捩じれてつながった (図 2.16) 螺旋状のトーラス空間である．また，全ての長調について，上が属調，下が下属調，左が平行調，左上が属調平行調，左下が下属調平行調，右が同主調というように関係調が隣接して並んでいる．短調も長調とは左右逆であるが同様である．

[3]その調が長調か短調かによって，左に置かれるか，右に置かれるかが変わる．長調ならば右に同主調をおく．これにより調性空間に同主調同士がペアとなり平行にならぶ．

level a:							*7*				
level b:			*2*				7				
level c:			*2*				7				*11*
level d:	0		2		4		*6*	7		9		11
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.10: $\delta(\mathrm{I}/\mathbf{C} \to \mathrm{I}/\mathbf{V}) = 1 + 1 + 5 = 7$

level a:			*2*								
level b:			2							*9*	
level c:	0		2				*6*			*9*	
level d:	0		2		4		6	7		9		11
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.11: $\delta(\mathrm{I}/\mathbf{C} \to \mathrm{V}^7/\mathbf{V}) = 1 + 2 + 7 = 10$

level a:					*4*						
level b:	0				*4*						
level c:	0				4				*8*		
level d:	0	*1*		*3*	4		*6*		*8*	9		
level e:	0	1	2	3	4	5	6	7	8	9	10	11

図 2.12: $\delta(\mathrm{I}/\mathbf{C} \to \mathrm{i}/\mathbf{c}\#) = 4 + 1 + 7 = 12$

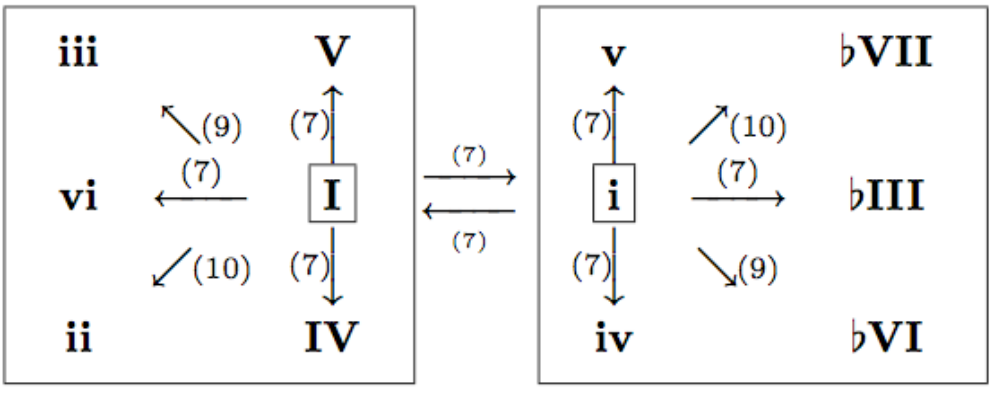

図 2.13: 近親調の距離と配置

d♯	**F♯**	**f♯**	**A**	**a**	**C**	**c**
g♯	**B**	**b**	**D**	**d**	**F**	**f**
c♯	**E**	**e**	**G**	**g**	**B♭**	**b♭**
f♯	**A**	**a**	**C**	**c**	**E♭**	**e♭**
b	**D**	**d**	**F**	**f**	**A♭**	**a♭**
e	**G**	**g**	**B♭**	**b♭**	**D♭**	**d♭**
a	**C**	**c**	**E♭**	**e♭**	**G♭**	**g♭**

図 2.14: 調性空間 (調名表記)

♯ii	**♯IV**	**♯iv**	**VI**	**vi**	**I**	**i**
♯vi	**VII**	**vii**	**II**	**ii**	**IV**	**iv**
♯i	**III**	**iii**	**V**	**v**	**♭VII**	**♭vii**
♯iv	**VI**	**vi**	**I**	**i**	**♭III**	**♭iii**
vii	**II**	**ii**	**IV**	**iv**	**♭VI**	**♭vi**
iii	**V**	**v**	**♭VII**	**♭vii**	**♭II**	**♭ii**
vi	**I**	**i**	**♭III**	**♭iii**	**♭V**	**♭v**

図 2.15: 調性空間 (相対音度表記)

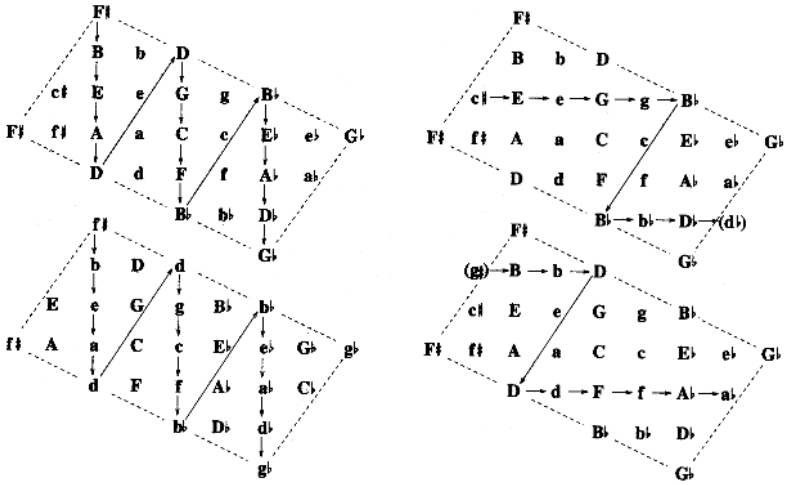

図 2.16: 調トーラスの繋がり ([9] pp.69)

この調性空間を用いて調間の距離ルールを定める．まず，ある調から移動できる範囲はその関係調のみであり，それらへの距離はそのトニック同士の距離とし，この範囲をピボット，その中心となる調をピボットのトニックとよぶ．遠隔調への距離も以下のようにして定める:

> **調間距離ルール**　$\Delta(\mathbf{I} \to \mathbf{R}) = [\delta_1(\mathbf{P_1} \to \mathbf{P_2})] + [\delta_2(\mathbf{P_2} \to \mathbf{P_3})] \ldots + \delta_n(\mathbf{P_n} \to \mathbf{R})$, ここで $\Delta(\mathbf{I} \to \mathbf{R})$ は現在のピボットのトニックである $\mathbf{I}$ から目標の調 $\mathbf{R}$ への距離; δ_1 は現在のピボットのトニック $\mathbf{P_1}$ から次のピボットのトニック $\mathbf{P_2}$ への距離 (他も同様); δ_n はピボットのトニック $\mathbf{P_n}$ から $\mathbf{R}$ への距離で，$\mathbf{R}$ がピボット内に入った時点で計算される．角括弧囲まれた部分は $\delta_n(\mathbf{P_n} \to \mathbf{R})$ の条件を満たしていない場合のみ適用される．

なお，このルールでは，$\Delta(x \to y)$ の値は経路の選択によって変化し，一意に定まらない．この問題については後述する (第 2.5 節).

調間距離の最小値と最短経路について幾つか例をあげる．

- $\Delta(\mathbf{C} \to \mathbf{g})$. $\mathbf{C} \xrightarrow{(7)} \mathbf{c} \xrightarrow{(7)} \mathbf{g} = 7 + 7 = 14$.
- $\Delta(\mathbf{C} \to \mathbf{b})$. $\mathbf{C} \xrightarrow{(9)} \mathbf{e} \xrightarrow{(7)} \mathbf{b} = 9 + 7 = 16$.
- $\Delta(\mathbf{E} \to \mathbf{F})$. $\mathbf{E} \xrightarrow{(7)} \mathbf{e} \xrightarrow{(9)} \mathbf{C} \xrightarrow{(7)} \mathbf{F} = 7 + 9 + 7 = 23$.
- $\Delta(\mathbf{C} \to \mathbf{F}\sharp)$. $\mathbf{C} \xrightarrow{(7)} \mathbf{c} \xrightarrow{(7)} \mathbf{E}\flat \xrightarrow{(7)} \mathbf{e}\flat \xrightarrow{(7)} \mathbf{F}\sharp = 7 + 7 + 7 + 7 = 28$.

$\Delta(\mathbf{C} \to \mathbf{F}\sharp)$(すなわち $\Delta(\mathbf{I} \to \sharp\mathbf{IV})$) の最小値は原著では 30 となっている．しかしながら例示したような経路により更に小さな値となることが本研究の過程で判明した．調間距離のルールと照らし合わせて考えるにこれは原著者の単純な計算ミスだと思われる．表 2.3 及び図 2.17 についても同様に本研究での検証結果を反映している．なおこの部分以外には検証結果との食い違いは生じなかった．

調 $\mathbf{I}$ から各調への Δ の最小値をまとめたのが，表 2.3 である．同様に調性空間に記したものが図 2.17 となる．

表 2.3: $\mathbf{I}$ を基準とした調間距離の最小値

調	**I**	**i**	**♭II**	**♯i**	**II**	**ii**	**♭III**	**♭iii**	**III**	**iii**	**IV**	**iv**
調間距離 Δ	0	7	23	23	14	10	14	21	16	9	7	14

調	**♯IV**	**♯iv**	**V**	**v**	**♭VI**	**♭vi**	**VI**	**vi**	**♭VII**	**♭vii**	**VII**	**vii**
調間距離 Δ	28	21	7	14	16	23	14	7	14	21	23	16

2.5 和音/調間距離

調間距離の計算は，和音間距離の特殊なケース (I/**I** からI/**R** への距離) であると考えることができる．そこで調間距離のルールを利用して以下のルールを定める．

> **和音／調間距離ルール**　$\Delta(\mathrm{C}_1/\mathbf{R_1} \rightarrow \mathrm{C}_2/\mathbf{R_2}) = [\delta_1(\mathrm{C}_1/\mathbf{R_1} \rightarrow \mathrm{I}/\mathbf{P_1})] + [\delta_2(\mathbf{P_1} \rightarrow \mathbf{P_2})] + [\delta_3(\mathbf{P_2} \rightarrow \mathbf{P_3})] \ldots + \delta_n(\mathrm{I}/\mathbf{P_n} \rightarrow \mathrm{C}_2/\mathbf{R_2})$, ここで $\delta_1(\mathrm{C}_1/\mathbf{R_1} \rightarrow \mathrm{I}/\mathbf{P_1})$ は $\mathbf{R_1}$ 内の C_1 から $\mathrm{C}_1/\mathbf{R_1}$ を含んだピボットのトニック $\mathbf{P_1}$ への距離; $[\delta_2(\mathbf{P_1} \rightarrow \mathbf{P_2})] + [\delta_3(\mathbf{P_2} \rightarrow \mathbf{P_3})] \ldots$ は仲介するピボットへの転換; $\delta_n(\mathrm{I}/\mathbf{P_n} \rightarrow \mathrm{C}_2/\mathbf{R_2})$ は $\mathrm{C}_2/\mathbf{R_2}$ を含んだピボットのトニック $\mathbf{P_n}$ から $\mathbf{R_2}$ 内の C_2 への距離である．(δ_1 は開始地点が $\mathrm{I}/\mathbf{P_1}$ であるとき適用されないため，オプションとなる)

原著では例として，$\Delta(\mathrm{vi}/\mathbf{F} \rightarrow \mathrm{V}/\mathbf{b})$ をあげている．このとき，**F** から **b** への調性空間における経路を $\mathbf{F} \rightarrow \mathbf{C} \rightarrow \mathbf{e} \rightarrow \mathbf{b}$ のようにとると，

$$\delta_1(\mathrm{vi}/\mathbf{F} \rightarrow \mathrm{I}/\mathbf{C}) = 1 + 2 + 7 = 10 \tag{2.1}$$

$$\delta_2(\mathrm{I}/\mathbf{C} \rightarrow \mathrm{i}/\mathbf{e}) = 9 \tag{2.2}$$

$$\delta_3(\mathrm{i}/\mathbf{e} \rightarrow \mathrm{V}/\mathbf{b}) = 1 + 2 + 8 = 11 \tag{2.3}$$

とそれぞれの間の距離が計算でき (図 2.5，図 2.5)，全体は

$$\Delta(\mathrm{vi}/\mathbf{F} \rightarrow \mathrm{V}/\mathbf{b}) = 10 + 9 + 11 = 30 \tag{2.4}$$

となる．

しかしながら，和音／調間距離ルールでは $\Delta(x \rightarrow y)$ はその経路として選択する調に依存し，一意には定まらない．そこで 次の**最短経路の原理**を定める．

> **最短経路の原理**　二つのイベントの間のピッチ空間距離はなるべく最小の値を取るよう計算する．

ここで言う「イベント」とはピッチ，和音，そして (暗黙的な) 調の発音であり,「ピッチ空間距離」はピッチ間距離，和音間距離，調間距離の総称である．

2.6 TPS の特性

以上が Lerdahl の和音間距離の概略である．本節では，これらをもとに TPS の持つ特性について論じたい．

TPS は以下のような好ましい性質を持っているといえる．

	VII(23)	**vii**(16)	**II**(14)				
♯**i**(23)	**III**(16)	**iii**(9)	**V**(7)	**v**(14)			
♯**iv**(21)	**VI**(14)	**vi**(7)	**I**(0)	**i**(7)	♭**III**(14)	♭**iii**(21)	♯**IV**(28)
		ii(10)	**IV**(7)	**iv**(14)	♭**VI**(16)	♭**vi**(23)	
			♭**VII**(14)	♭**vii**(21)	♭**II**(23)		

図 2.17: 調性空間への距離の配置

vi/**F**

		2									
		2							9		
0		2			5				9		
0		2		4	5		7		9	10	
0	1	2	3	4	5	6	7	8	9	10	11

I/**C**

0											
0							*7*				
0				*4*			*7*				
0		2		4	5		7		9		*11*
0	1	2	3	4	5	6	7	8	9	10	11

図 2.18: $\delta(\text{vi}/\mathbf{F} \to \text{I}/\mathbf{C}) = 1 + 2 + 7 = 10$

i/**e**

				4							
				4							11
				4			7				11
0		2		4		6	7		9		11
0	1	2	3	4	5	6	7	8	9	10	11

V/**b**

						6					
	1					*6*					
	1					*6*				*10*	
	1	2		4		6	7			*10*	11
0	1	2	3	4	5	6	7	8	9	10	11

図 2.19: $\delta(\text{i}/\mathbf{e} \to \text{V}/\mathbf{b}) = 1 + 2 + 8 = 11$

1. 和声学との近似．TPS で計算される距離は和声学的な作曲規則，しいては人間の音楽認知との近似が見られる．上記の例であげたように，和音間距離は五度音，四度音に対して小さい値を示し，全体としてはカデンツ規則に類似している．調においても，近親調，特に属調・下属調・並行調・同主調が近く配置されており，遠隔調ほど遠くになる．

2. 距離の定量化．和音間距離・調間距離の関数は計算結果として整数値が返される．これは，二つ距離の比較し，どちらの進行がより適切であるかを選択できるということを意味する．これは従来の研究では困難であった．

3. 全域的アルゴリズム．和音間距離ルール・調間距離ルールは，引数として如何なる和音でもとることができるという点において，全域的なアルゴリズムである．構成音が複雑な和音に対しても，ベーシックスペースにより他の和音との差異を記述でき，それにより計算可能である．調についても同様であり，一般に用いられる西洋音楽由来の自然的長音階・和声的短音階に限らず，日本の古典芸能で用いられる律旋法などにも対応できる可能性を秘めている．

4. シンプルさ．距離のルールは，和音の五度圏・調の五度圏，そしてベーシックスペースで表される構成音の差異という既知の音楽理論的要素の組み合わせにより成り立っている．また計算量の面でも，次章のように実装方法を工夫することにより定数時間アルゴリズムとなっている．これはシステムに組み込み応用研究を行う上で都合がよい．

しかしながら，幾つかのデメリットも存在する．

1. 曖昧性と定義の不備．TPS は，和音の認知に対する定量化を行うという点で新規性と独創性のある理論であり，興味深い研究である．しかし，細部には曖昧性を含んだ表現や厳密に考慮すると矛盾する部分が見られ，そのままでは実際に計算機により実装することは困難である．本稿では次章にて実装方法を検討し，この問題の解決を試みる．

2. 交換法則．和音間距離には交換法則が成り立つ ($\delta(x \to y) = \delta(y \to x)$)．これは一見するとメリットであるが，和声学と照らし合わせたときに問題が生じる．例えば，$\mathrm{I} \to \mathrm{IV} \to \mathrm{V} \to \mathrm{I}$ という和声進行は和声学のカデンツ規則により許容される**よい**進行であるが，その逆の進行 $\mathrm{I} \to \mathrm{V} \to \mathrm{IV} \to \mathrm{I}$ はカデンツ規則から外れた**悪い**進行である．しかし，和音間距離の総和としては両者は等しくなってしまう．これは和声学と TPS が完全な一致をするわけではないことを示す．

最後の点について本研究ではそれほど大きな問題とは考えない．本研究が目的としているのは楽曲からの和声解析であり，カデンツ規則から外れた進行に対しても解を出せるのはむしろ利点である．この問題が重要となるのは，本研究のような解析的分野ではなく，TPS を利用して新しい楽曲を作るなどの生成的な分野であると言える．また，本稿では

扱わなかったがTPSでも緊張値 (tension value) の計算には向きが生じて交換法則は成り立たない．

以上をまとめると，TPSはシンプルで全域的な人間の楽曲認知を定量化するアルゴリズムだと言える．この理論は，和声学の和音に対する見解と等しく，和声学より広い範囲で和声進行について論じることができるという点で和声解析にとって有望なツールである．しかしながら，細部には曖昧性が残り，実装には気を使わなければならない．また，楽曲生成などに用いるには何らかの工夫が必要であると言える．

第3章　TPSの整備

本章では，TPS の定義の修正と計算機での実装について述べる．

3.1　実装における問題点

TPS は聞き手の心理の量化を行うという新規性のあり応用が期待される音楽理論であるが，これを計算機上で実装するには幾つかの問題を解決する必要がある．その問題とは以下のようなものである．

- ピッチ・調・和音といった音楽要素の定義が不十分である．特に，調と和音の関係が曖昧で，時に混同されているため，ルールの理解が難しくなっている．
- ルール中に示される数式について，一般的ではない表記が用いられている．また，経路の選択など不明瞭な部分がある．$\delta(x \to y)$ や $\Delta(x \to y)$ は経路を参照しており厳密には関数と言えない．
- 計算機上ではより単純な構造により実装できる部分がある．

そこで本研究では原著のルールが意図するところを極力そのままに，計算機での実装が容易となるように，TPS の定義に修正を加える．

3.2　ピッチクラス

ピッチクラスは 0 から 11 の整数値であり (表 3.1)，それぞれが音名と対応している．ただし，音名からピッチクラスへの変換は容易に行えるのに対して，ピッチクラスから音名への変換をピッチクラスのみから行うことはできない．ピッチクラス 1 が C♯ と D♭ の二つの名前を持っているように，各ピッチクラスがどの同音異名で用いられるかは，その音が置かれている調によって変わるからである．

表 3.1: ピッチクラス型

型	内容
整数値 $(0 \ldots 11)$	ピッチクラス値

また，あるオクターブのBの次の音高が一つ上のオクターブのCであることから，ピッチ・クラスは循環構造を持っていると言える．よって，あるピッチ・クラス値nからmだけ高い音程のピッチ・クラス値は$(n+m) \bmod 12$で求めることができる．

3.3 調

調は半音階の中から相性の良い音の組み合わせを抜き出してできるグループであり，音階の基礎となる音程 (主音) のピッチ・クラスと，長音階・短音階といった音階の種類との組によって表現できる (表 3.2)．本研究では，一般的に広く用いられる長音階と短音階 (和声的短音階) に限定して考え，主音 12 のそれぞれに長調・短調が存在するため 24 の調を対象とする．

表 3.2: 調構造体のメンバー

アクセス子	型	内容
root	ピッチクラス	調の主音となるピッチクラス
scale	長調 or 短調	音階の分類

各調の音階に含まれる音を調の構成音とよぶ．また，構成音には主音から高い音へと順番に，I から VII のローマ数字が振られる．これを音度とよぶ．調の構成音以外についても♭II のように音度で表すことが可能である．なお，一般的な音楽の教科書では音度についても調における長短により大文字・小文字を使い分けて表記されることが多いが，本提案システムではこのような表記はとらなかった．音度の長短は調により自明だからである．これは単に表記の問題であり，内部的には長短が区別されている．

長調**C**の構成音のピッチ･クラス値は$\{0, 2, 4, 5, 7, 9, 11\}$，短調**c**の構成音は$\{0, 2, 3, 5, 7, 8, 11\}$である．他の調については，**C**或いは**c**の構成音を，その調の主音のピッチ・クラス値だけシフトすることにより得ることができる (表 3.3)．

本稿では原著にならい，ヘ長調 (F Major) を “**F**”，ハ短調 (C Minor) を “**c**” のように，太字体のアルファベットで，長調ならば大文字，短調ならば小文字で，調名を表記する．また，ある調を中心とした相対的な表記として音度を利用して，**II**(**C**を中心とした時**D**) や **vi**(同様に **a**) のようにも書く．

3.4 和音とベーシックスペース

和音は音高の異なるいくつかの音の集まりであり，TPS では和音の調，音度，構成音，ベーシックスペースという四つの性質を参照する．調はその和音が置かれている調であり，全ての和音は何らかの調に属しているものと考え，その構成音は基本的に調の構成音から選択される．音度は根音が調の構成音を主音から昇順に並べたときに何番目の音であ

表 3.3: 構成音と音度

調名	0	1	2	3	4	5	6	7	8	9	10	11
C	I		II		III	IV		V		VI		VII
C♯	VII	I		II		III	IV		V		VI	
D		VII	I		II		III	IV		V		VI
D♯			VII	I		II		III	IV		V	
E				VII	I		II		III	IV		V
F	V		VI		VII	I		II		III	IV	
F♯		V		VI		VII	I		II		III	IV
G	IV		V		VI		VII	I		II		III
G♯	III	IV		V		VI		VII	I		II	
A		III	IV		V		VI		VII	I		II
A♯	II		III	IV		V		VI		VII	I	
B		II		III	IV		V		VI		VII	I
c	I		II	III		IV		V	VI			VII
c♯	VII	I		II	III		IV		V	VI		
d		VII	I		II	III		IV		V	VI	
d♯			VII	I		II	III		IV		V	VI
e	VI			VII	I		II	III		IV		V
f	V	VI			VII	I		II	III		IV	
f♯		V	VI			VII	I		II	III		IV
g	IV		V	VI			VII	I		II	III	
g♯		IV		V	VI			VII	I		II	III
a	III		IV		V	VI			VII	I		II
a♯	II	III		IV		V	VI			VII	I	
b		II	III		IV		V	VI			VII	I

るかを意味し，I〜VII のローマ数字で表記する．これは和音の機能と対応している．構成音は和音に含まれている音であり，含まれる音度によりトライアドやセブンス・コードなどの和音に分類できる．本研究では，ハ長調 (**C**) の I 番目の和音であれば “I/**C**” のように，ローマ数字表記による音度と太字のアルファベット表記の調の間にスラッシュを挿んだものによって和音を記述する．ただし，文脈上で調が自明である場合には，調を省略し単に “I” のように表記する．また，和音の種類に応じて，一般的なコードネームの表記に従って添字を挿む．セブンス・コードであれば，“$\mathrm{V}^7/\mathbf{C}$” のようになる．

表 3.4: 和音構造体のメンバー

アクセス子	型	内容
key	調構造体	和音のおかれている調
root	ピッチクラス型	和音の根音となるピッチクラス
notes	真偽値配列 [12]	各ピッチクラスが発音されていれば真
basicspace	整数値配列 [12]	上三要素から生成されるベーシックスペース

距離の計算で使用されるベーシックスペースは和音の上記とは異なる表現方法であり，調・音度・構成音から生成可能である．ベーシックスペースは和音を構成する音に重み付けを行った構造体であり，2.2 節で述べた性質より，各ピッチクラスが和音に与えている影響度とみなして図 3.1 のように棒グラフで表せる．計算機上の実装としては

$$\mathrm{I}/\mathbf{C}.basicspace = [4, 0, 1, 0, 2, 1, 0, 3, 0, 1, 0, 1] \tag{3.1}$$

のような，長さ 12 で添字としてピッチクラスをとる配列となる．

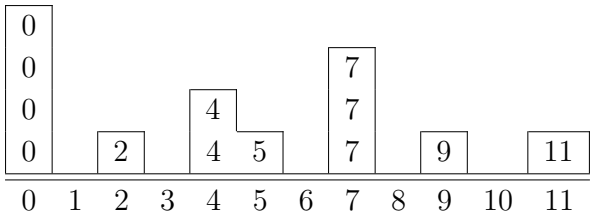

図 3.1: I/**C** の棒グラフ表現

原著では和音の五度圏や全音階の五度圏をベーシックスペース上のオペレーションとして定義し，和音の一次の表現方法としてベーシックスペースを使用しているが，本研究では和音を調・音度・構成音という三つの要素を持った構造体として実装し，そこからベーシックスペースを生成するという手法をとった．ベーシックスペースには調についてその構成音の情報のみが含まれており，並行調同士は同じ構成音をもっているため，ベーシックスペースのみからは調を特定することが不可能だからである．ベーシックスペースを一次の和音表現とした場合，この点が調間距離の計算において問題となる．

3.5 和音間距離

続いて，完全版の和音ルールを実装する．完全版の和音ルールは，実際には二つの和音の置かれている調が近親調の関係にある場合のみ適応される．和音間距離は，i, j, k という三つの観点から求められた数値の和である．また，i, j, k はそれぞれ和音 x と y に依存した値であり，x, y を引数とする関数としてとらえることができる．本研究ではこの点を明示するため，近親調の和音間距離関数を $\delta_{near}(x, y)$ とよび，以下のように定義した:

$$\delta_{near}(x, y) = \mathrm{region}(x, y) + chord(x, y) + basicspace(x, y). \tag{3.2}$$

3.5.1 調の五度圏における距離

$region(x, y)$ は調の五度圏における距離を求める関数である．この関数は直観的には調の五度圏の円において二つの調がどれだけ離れた弧に存在するかで考えることができる．原著ではベーシックスペース上のオペレーションによって定義されていたが，これは以下のような調のピッチクラスを添字とするテーブル参照として実装することができる．なお，このテーブルの添字は長調のピッチクラスであるため，短調の和音のピッチクラスは，その平行調のピッチクラスに変換する必要がある．

表 3.5: 調の五度圏テーブル RegionCircle

pc	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*
0	0	5	2	3	4	1	6	1	4	3	2	5
1	5	0	5	2	3	4	1	6	1	4	3	2
2	2	5	0	5	2	3	4	1	6	1	4	3
3	3	2	5	0	5	2	3	4	1	6	1	4
4	4	3	2	5	0	5	2	3	4	1	6	1
5	1	4	3	2	5	0	5	2	3	4	1	6
6	6	1	4	3	2	5	0	5	2	3	4	1
7	1	6	1	4	3	2	5	0	5	2	3	4
8	4	1	6	1	4	3	2	5	0	5	2	3
9	3	4	1	6	1	4	3	2	5	0	5	2
10	2	3	4	1	6	1	4	3	2	5	0	5
11	1	2	3	4	1	6	1	4	3	2	5	0

ある短調からその並行調への変換は主音を三つ高い音へとシフトすることにより実現できる．以上より，近親調の和音 x と y の和音間距離関数 $region(x, y)$ は図 3.5 のテーブル $RegionCircle$ と和音型を長調のピッチクラスに変換する関数 $pcm(x)$ を用いて，

$$region(x, y) = RegionCircle[pcm(x)][pcm(y)] \tag{3.3}$$

と書ける．また，$pcm(x)$ は $x.pc$ をその和音のおかれている調のピッチクラス，$x.scale$ をその和音の調が長調であるかの真偽値として，

$$pcm(x) = \begin{cases} x.pc & (x.scale = \text{長調 のとき}) \\ (x.pc + 3) \bmod 12 & (x.scale = \text{短調 のとき}) \end{cases} \tag{3.4}$$

により定義できる．

3.5.2 和音の五度圏における距離

同様に $chord(x, y)$ は和音の五度圏における距離を求める関数である．こちらも和音の五度圏の円によりその概要を図示することができる．調の五度圏において二つの調がどれだけ離れているかを定量化したのに対し，和音の五度圏は二つの和音の音度がどれだけ離れているかを弧の数によって数値化する．ここで問題となるのが二つの和音がことなる調に置かれている場合で，このとき，両者の音度は基準となるピッチが異なるため一概に比較することはできない．原著では，このようなケースにおける比較方法について明確に述べられていないが，幾つかの例より一方の調を基準として他方の音度を変換していることがみてとれる．そこで本研究では，次のようなルールにより音度の変換を行うこととした．

1. 和音 x の音度を和音 y の調上での音度に変換する (x').
2. 和音 y の音度を和音 x の調上での音度に変換する (y').
3. x' と y の音度と，x と y ダッシュの音度をそれぞれ和音の五度圏で比較し，短い方を取る．

ただし，全ての和音が全ての調に変換できるわけではない．例えば，I/**C** と I/**C**♯ を比較しようとしたとき，**C** と **C**♯ のどちらも音度の変換が行えない．このとき，変換が不可能だった和音には便宜的に音度 0 が割り当てられる．音度 0 との和音の五度圏における距離の計算は十分に大きな数となるようにした．実装では 99 とした．$\delta(x, y)$ は “良い” 進行ならば 10 以下，最悪の場合でも 40 程度となるので，この値は十分と考える．これは，二つの和音の音度・調の割当が不当であり，より妥当な解釈が存在することを示す．以上をふまえて和音の五度圏をテーブル化したものを表 3.6 に記す．

また，$chord(x, y)$ は次のように定義できる:

$$\text{chord(x,y)}=\begin{cases} ChordCircle[x][y'] & (ChordCircle[x][y'] \leq ChordCircle[x'][y]) \\ ChordCircle[x'][y] & (ChordCircle[x][y'] > ChordCircle[x'][y]) \end{cases} \tag{3.5}$$

3.5.3 ベーシックスペースにおける違いの距離

最後にベーシックスペースの比較は以下の式によって定義した:

$$basicspace(x,y) = \frac{\left(\sum_{i=0}^{11} |x.basicspace[i] - y.basicspace[i]| + 1\right)}{2}. \tag{3.6}$$

これは原著に準拠したものであり，ピッチクラス毎に差を求め合計することにより計算を簡略化している．$+1$ は総和が 2 で割り切れない場合に端数を切り上げることを示す．これは，三和音とセブンス・コードのように構成音の数が違う和音間で比較を行ったときのための処理で，原著の例と一致させるためにおこなっている．

3.6 調間距離

続いて，調 r_1 と調 r_2 の間の距離関数 $\Delta(r_1, r_2)$ の定義を行う．原著では，$\Delta(x, y)$ を結果が集合となる式として使用していたが，本研究では和音と調を異なる型のものと区別し，調の演算に対して Δ を使用することとした．

調間距離は第 2.4 節の調性空間を用いて計算される．また，調性空間は，24 の調をそれぞれノードとし，近親調の間にエッジを結んだ無向グラフとして表現できる．このとき，エッジの長さは，両端の調の間の関係によって表 3.6 のように決定される．

ある二つの調が近親調の関係にないとき，その距離はグラフ上での最短距離によって求められる．よって，二つの調の間の距離は和音の音度によらず，調性空間グラフより静的に求められるといえる．そこで本研究では，調性空間グラフから距離を事前に計算しておき，テーブル参照する実装をとった．調性空間グラフを作成し，ダイクストラ法によって各調間の距離を求めた結果が表 3.8 である．

以上をまとめると，調 r_1 から調 r_2 への距離は調間距離テーブル R を用いて次のように定義できる:

$$\Delta(r_1, r_2) = Region[toindex(r_1)][toindex(r_2)] \tag{3.7}$$

ここで，関数 $toindex(r)$ は調 r をインデックスに変換するもので，

$$toindex(r) = \begin{cases} r.pc & (r.scale = \text{長調}) \\ r.pc + 12 & (r.scale = \text{短調}) \end{cases} \tag{3.8}$$

により定義される．

また，近親調・遠隔調の関係について調間距離をもちいて定義し直すことができる．すなわち，調 R_1 と調 R_2 が $\Delta(R_1, R_2) \leq 10$ を成立させるとき，それを近親調とよび，そうでない場合を遠隔調とよぶ．

3.7　修正和音間距離

ここまでで定義した $\delta_{near}(x,y)$ と $\Delta(r_1,r_2)$ を用いて，遠隔調にも適用可能な和音間距離関数 $\delta(x,y)$ を定義する．$\delta(x,y)$ は和音 x と y が近親調にあるときは，$\delta_{near}(x,y)$ と等しい．また，両者が遠隔調の関係にあるときは，x と近親調にあるトニックの和音 $\mathrm{I}/\mathbf{R_1}$ と，y と近親調にあるトニックの和音 $\mathrm{I}/\mathbf{R_2}$ を置くことにより，x から $\mathrm{I}/\mathbf{R_1}$，$\mathbf{R_1}$ から $\mathbf{R_2}$，$\mathrm{I}/\mathbf{R_2}$ から y の和音間距離・調間距離の和である．ただし，このとき，$\mathbf{R_1}$ と $\mathbf{R_2}$ は和が最小となるように選択されなければならない．

よって $\delta(x,y)$ は，その和音の近親調内にあるトニックの集合を求める関数 $\mathrm{P}(x)$ をもちいて，x と y の関係により場合分けして，

$$\delta(x,y)=\begin{cases} region(x,y)+chord(x,y)+basicspace(x,y)=\delta_{near} & (\text{近親調}) \\ \min\left\{\delta(x,\mathrm{I}/\mathbf{R_1})+\Delta(\mathbf{R_1},\mathbf{R_2})+\delta(\mathrm{I}/\mathbf{R_2},y)|\mathbf{R_1}\in\mathrm{P}(x),\mathbf{R_2}\in\mathrm{P}(y)\right\} & (\text{遠隔調}) \end{cases} \tag{3.9}$$

のように再帰的に定義できる．各調に対する近親調は自身を含めて 7 つ存在するので，遠隔調の場合には $7^2=49$ の経路について探索することとなる．

表 3.6: 和音の五度圏テーブル ChordCircle

音度	0	I	II	III	IV	V	VI	VII
0	99	99	99	99	99	99	99	99
I	99	0	2	3	1	1	3	2
II	99	2	0	2	3	1	1	3
III	99	3	2	0	2	3	1	1
IV	99	1	3	2	0	2	3	1
V	99	1	1	3	2	0	2	3
VI	99	3	1	1	3	2	0	2
VII	99	2	3	1	1	3	2	0

表 3.7: 調の関係と距離

関係	並行調	属調・下属調	同主調	属調並行調	下属調並行調
距離	7	7	7	9	10

表 3.8: 調間距離テーブル Region

調	C	C♯	D	D♯	E	F	F♯	G	G♯	A	A♯	B	c	c♯	d	d♯	e	f	f♯	g	g♯	a	a♯	b
C	0	23	14	14	16	7	28	7	16	14	14	23	7	23	10	21	9	14	21	14	23	7	21	16
C♯	23	0	23	14	14	16	7	28	7	16	14	14	16	7	23	10	21	9	14	21	14	23	7	21
D	14	23	0	23	14	14	16	7	28	7	16	14	21	16	7	23	10	21	9	14	21	14	23	7
D♯	14	14	23	0	23	14	14	16	7	28	7	16	7	21	16	7	23	10	21	9	14	21	14	23
E	16	14	14	23	0	23	14	14	16	7	28	7	23	7	21	16	7	23	10	21	9	14	21	14
F	7	16	14	14	23	0	23	14	14	16	7	28	14	23	7	21	16	7	23	10	21	9	14	21
F♯	28	7	16	14	14	23	0	23	14	14	16	7	21	14	23	7	21	16	7	23	10	21	9	14
G	7	28	7	16	14	14	23	0	23	14	14	16	14	21	14	23	7	21	16	7	23	10	21	9
G♯	16	7	28	7	16	14	14	23	0	23	14	14	9	14	21	14	23	7	21	16	7	23	10	21
A	14	16	7	28	7	16	14	14	23	0	23	14	21	9	14	21	14	23	7	21	16	7	23	10
A♯	14	14	16	7	28	7	16	14	14	23	0	23	10	21	9	14	21	14	23	7	21	16	7	23
B	23	14	14	16	7	28	7	16	14	14	23	0	23	10	21	9	14	21	14	23	7	21	16	7
c	7	16	21	7	23	14	21	14	9	21	10	23	0	23	14	14	16	7	28	7	16	14	14	23
c♯	23	7	16	21	7	23	14	21	14	9	21	10	23	0	23	14	14	16	7	28	7	16	14	14
d	10	23	7	16	21	7	23	14	21	14	9	21	14	23	0	23	14	14	16	7	28	7	16	14
d♯	21	10	23	7	16	21	7	23	14	21	14	9	14	14	23	0	23	14	14	16	7	28	7	16
e	9	21	10	23	7	16	21	7	23	14	21	14	16	14	14	23	0	23	14	14	16	7	28	7
f	14	9	21	10	23	7	16	21	7	23	14	21	7	16	14	14	23	0	23	14	14	16	7	28
f♯	21	14	9	21	10	23	7	16	21	7	23	14	28	7	16	14	14	23	0	23	14	14	16	7
g	14	21	14	9	21	10	23	7	16	21	7	23	7	28	7	16	14	14	23	0	23	14	14	16
g♯	23	14	21	14	9	21	10	23	7	16	21	7	16	7	28	7	16	14	14	23	0	23	14	14
a	7	23	14	21	14	9	21	10	23	7	16	21	14	16	7	28	7	16	14	14	23	0	23	14
a♯	21	7	23	14	21	14	9	21	10	23	7	16	14	14	16	7	28	7	16	14	14	23	0	23
b	16	21	7	23	14	21	14	9	21	10	23	7	23	14	14	16	7	28	7	16	14	14	23	0

第4章　提案手法と実装

本章では，本研究で作成した TPS を利用した和声進行解析手法について概略とその実装について述べる．

4.1　提案システム概要

提案システムは，入力としてコードネームの配列を受け取り，それぞれのコードネームに対して適切な和音を認識し出力することを目的とする．入力として受け取るコードネームの配列とは，楽曲から和音区間と構成音を認識し，それらを主音及び和音の型 (三和音，セブンス・コードなど) により分類したものの列である．それに対して，ここで認識する和音とは配列の各要素が，楽曲中でどの調におかれ，どの音度 (役割) を持っているかと考えることができる．役割と調は，その対象とする部分のコードネームのみから一意に決定することはできない．これは和音の構成音からは，複数の調が考慮可能だからである．しかしながら，コードネーム (構成音)，音度 (根音)，調のうち二つが確定できれば残りの一つの要素についても一意に確定することができる．

そこで，本研究では構成音に対して，適合する調を列挙し，それらの調の中でどれが最も相応しいかを TPS による和音間距離を基準として選択する方法を提案する．構成音が与えられたとき，その置かれ得る調はスケールアウトするかどうかをみることにより，ある程度絞り込むことができる．その時，絞り込まれた調から音度についても推定可能となる．よって，和声の解析はいくつかの調の中からどの調が一番妥当であるかを検討することに帰着できる．提案手法では調を検討する手段として TPS の和音間距離を使用し，楽曲全体の和音間距離総和が最も小さくなる進行を最適解として提示する．

提案システムは和声候補生成・和声進行グラフ生成・TPS 距離計算・最短距離計算の四つの過程に分かれ，解析がおこなわれる．

以下では，提案システムの各部位について説明する．

4.1.1　和音候補生成部

和音候補生成部は，与えられた構成音データから各コードネームに対して考えられる音度と調を候補として列挙し，それらを元にベーシックスペースを作成，和音インスタンスを生成する．この部分は更に，コードネーム・構成音相互変換部，調候補列挙部，和音候補生成部の三つに分かれる．

4.1.1.1 コードネーム・構成音相互変換部

コードネーム・構成音相互変換部は，コードネームと構成音の相互変換をおこなう．これらは和音候補の生成に必要な要素である．

4.1.1.2 調候補列挙部

調候補列挙部は，構成音を受け取り，その構成音と各調の構成音を見比べることで，和音が置かれうる調を列挙する．

4.1.1.3 和音候補生成部

和音候補生成部は，上記より得られた和音情報と調候補を組み合わせ和音候補を生成する．

4.1.2 TPS 部

TPS は，二つの和音候補を受け取りその和音間距離を計算する．

4.1.3 和声進行グラフ生成部

和音進行グラフ生成部は和音候補生成部より，和音インスタンスの集合のリストを受け取り，それを基に和声進行グラフを生成して，最短距離計算部へと渡す．

和音進行グラフは，和音候補をノード，譜面上で隣接する和音候補間をその和音間距離を値とするエッジで結んだグラフである．

4.1.4 最短距離計算部

最短距離計算部では，和声進行グラフ生成部より受け取った和声進行グラフより最短経路を計算して，それを最適解として出力する．

4.2 借用和音の扱いについて

楽曲中ではしばしば現在の調性から外れた和音が導入されることがある．このような和音は他の調から借りてきた和音として扱われるが，本研究では，借用和音という考えは導入せずにそれらの部分について短い転調がおこなっていると見なした．借用和音は聴衆がその調性を感じることができないほどに短い区間の転調であると考えられるからである．従来研究では，短い転調を考えることは，解析時にサンプルとして少なすぎるためノイズ

と見なされてしまうなどの理由により難しかったが，TPS は転調として考えた方が解析が行いやすい．また，カデンツの終止形の理解の上でも借用和音を短い転調と見なす方が都合が良い．短い転調という概念はあまり一般的ではないが，借用和音と短い転調は比較的容易に相互変換可能であり，特に問題はないと考える．

4.3 実装

提案システムの実装は java で行い，入力ファイル形式としてコードネームを記したプレインテキストを，出力ファイル形式として音度と調を記したプレインテキストと，和声候補と最適解となった進行を示した graphviz 形式のグラフを用いた．以下に，大まかなクラスの実装について記す．なお TPS により和音間距離を計算する部分については第 3 章で述べた通りである．この部分については説明を省略する．また，実装はオブジェクト指向を考慮しておこなった．そのため第 3 章の内容と順序・配置が異なる箇所が存在する．

4.3.1 PitchClass

クラス PitchClass は，各々のピッチクラスを示す簡易なクラスであり，メンバーとしてそのピッチクラスの整数値を持つ．またピッチクラスの音名への変換を行うメソッド toStringMajor() と toStringMinor() を持つ．両者の違いは長音階において一般的に使用される音名を出力するか，短音階において一般的に使用される音名を出力するかである．このクラスは実質的に整数値であるが，ピッチクラスが持つ特性 (環状であることなど) を考慮し，整数を引数とする演算を避けるために別のクラスとした．

このクラスは後述のクラス KeySymbol および ChordSymbol で主として和音名の変換のために使用される．また，このクラスからのオブジェクト生成・管理は次節の PitchClass-Manager に一任される．

4.3.2 PitchClassManager

クラス PitchClassManager は，PitchClass インスタンスの生成・管理を行うクラスである．このクラスは PitchClass インスタンスの生成を制限するため Singleton と Flyweight のデザインパターン [3] を適用している．

このクラスの主たる役割は，ピッチクラス値或いは音名から PitchClass インスタンスを検索し返すことである．ピッチクラスを示す整数値からの変換は第 2.2 節の表 2.1 により行われる．音名 (文字列) よりの変換は連想配列 (ハッシュテーブル) によるテーブル参照である．なお，ピッチクラス値のシフトは第 3.2 節で述べたように $(x + y) \bmod 12$ で行うことができるが，java をはじめとした多くの言語に備わっている余算関数%は結果が負の数になりえる剰余算である．TPS ではこれらの組み込み関数ではなく結果が $0 \ldots 11$ の範囲に収まる数学的定義の余算を用いる必要がある．

4.3.3 KeySymbol

クラス KeySymbol は調を示すクラスであり，各インスタンスは調の主音となる PitchClass と調の音階の種別を情報として持ち，それらから得られる各種の調情報を提示する．音階の種別は，長音階ならば真，和声的短音階ならば偽の真偽値により実装した．これは主として使われる音階がこの二つであることからの選択であるが，将来的な拡張としては，列挙型で値をもち自然的短音階などの他の音階にも対応することが考えられる．

第一の用途は，ある PitchClass インスタンスがその調において第何度の音であるかを返すものである．これは第 3.3 節の表 3.3 を参照しており，ベーシックスペースの作成する際に五度音を特定するためなどに利用される．第二の用途は，調間の関係・距離の提示である．和音間距離の計算における調の五度圏での距離，及び調間距離の計算を行い，その結果を返す．

このクラスも PitchClass の場合と同様に次節の KeyManager により生成・管理される．

4.3.4 KeyManager

クラス KeyManager は，KeySymbol インスタンスの生成と管理を行うクラスであり，PitchClassManager 同様，Singleton と Flyweight パターンが適応されている．このクラスは各ピッチクラスの長調・短調に対応する KeySymbol を生成し，その管理と受け渡しを行う．

このクラスでは，PitchClassManager と同様に，主音と音階種別，或いは調名からの KeySymbol の検索を他に，近親調の提示と，和音の構成音からの調候補の提示を行う．

4.3.5 Basicspace

クラス Basicspace はベーシックスペースを表現しており，主に後述の ChordSymbol のメンバーとして用いられる．Basicspace は長さ 12 の整数配列をメンバーとしてもっている．

また，今後の拡張性のため複数のコンストラクタを所持している．インスタンスの生成に主として用いられるのはコンストラクタ Basicspace(KeySymbol key, PitchClass root, boolean[] notes) である．これはその和音が置かれている調 key(KeySymbol インスタンス) と，和音の根音ピッチ root(PitchClass インスタンス)，そして各ピッチクラスが和音の構成音であるかの真偽値配列 notes の三つを引数としてとり，そこからベーシックスペースを構築する．

Basicspace は，根音のピッチクラスを返すメソッド rootPitchClass() と，他の Basicspace を引数にとりベーシックスペースの違いの数を計算するメソッド distance(Basicspace bs2) をもつ．前者はベーシックスペースと調から ChordSymbol を生成する際に，後者は和音間距離を求める際に使用される．

4.3.6 ChordSymbol

ChordSymbol は TPS における和音を実装するクラスである．このクラスはメンバーとして，Basicspace と KeySymbol をもっている．また，これについても Basicspace と同様に，今後の拡張性を考慮し和音インスタンスの生成方法が複数用意されている．

4.3.7 ChordDiscriminator

クラス ChordDiscriminator は，コードネームと構成音の相互変換を行うクラスである．このクラスはコードネームとその構成音を記した CSV ファイルを読みこみ，コードネームの文字列を受け取りその構成音の真偽値配列を返すメソッドと，構成音を示す浮動小数点配列を受け取りそのコードネームを返すメソッドを提供する．

コードネーム／構成音変換用 CSV ファイルは図 4.1 のように記載されており，手前から 12 のフィールドがそれぞれピッチクラスに対応し，最後のフィールドが和音名となっている．構成音のフィールドは浮動小数点配列に変換されて保持している．これは，後述する構成音からの変換のためである．

```
1,1,0,0,1,0,0,0,1,0,0,0,C#mM7
0,1,0,0,0,1,0,0,1,0,0,1,C#7
1,0,1,0,0,1,0,1,0,0,0,0,G7sus4
1,0,1,0,0,1,0,0,1,0,0,0,Dm7-5
1,0,1,0,0,1,0,0,1,0,0,0,Fm6
1,0,1,0,0,1,0,0,0,1,0,0,Dm7
1,0,1,0,0,1,0,0,0,1,0,0,F6
1,0,1,0,0,0,0,1,0,1,0,0,D7sus4
1,0,1,0,0,0,0,1,0,0,0,0,C9
1,0,1,0,0,0,0,1,0,0,0,0,Gsus4
1,0,0,1,0,1,0,0,1,0,0,0,Fm7
1,0,0,1,0,1,0,0,1,0,0,0,G#6
1,0,0,1,0,1,0,0,0,0,1,0,F7sus4
...
```

図 4.1: コードネーム／構成音変換 CSV の一部

コードネームから構成音への変換は単純なルックアップであり全ての行に和音名がマッチしない場合には例外が返される．

構成音からコードネームへの変換は，コードネーム／構成音のデータベースとの比較により行われる．比較の方法としては最近傍法を使用している．あえて真偽値配列の比較にしなかったのは，構成音としてアナログな値を許すことで非和声音への対応を考慮してのものである．

4.3.8 ChordAnalyzer

クラス ChordAnalyzer は，和音候補グラフの生成，最短経路計算，及び和声進行データの出力を担当するクラスである．

4.3.8.1 和音候補グラフの生成

和音候補グラフの生成は，和音候補のセットを受け取ることによりインタラクティブに行われる．

まず，初期状態として ChordAnalzer インスタンスが生成された際に，グラフ領域が確保され，そこに起点ノードが一つがおかれる．グラフのノードは ChordSymbol インスタンスであるが，起点ノードは特殊な ChordSymbol として扱われ，例外的に他のノードへの距離 (エッジの値) は 0 に設定されている．すなわち，ChordAnalyzer インスタンス一つにつき一つの和音候補グラフを生成することになり一つの入力データに対応する．また，このときに最近の追加ノード集合として起点ノードが保持される．

続いて，ChordAnalyzer インスタンスはメソッド appendNodeSet(Set ¡ChordSymbol¿ nodes) を通じて，続くイベントに対する和音候補のセットを受け取る．そして，受け取った和音候補のセットをグラフに追加し，それら各々と最近の追加ノード (一つ前のイベントに対する和音候補) 全てとの間にエッジを繋ぎ，長さとして和音間距離を計算する．また，入力として受け取った和音候補セットを新しい最近の追加ノード集合として更新する．すなわち，ChordAnalyzer は楽譜上の並び通りに左から右への形式で和音候補を受け取り，随時処理していくこととなる．なお，グラフ上で非同一のものとして扱うために，同じ和音候補であっても異なるイベントに対する候補である場合には別のインスタンスを生成する必要がある．

この作業をイベント列 (提案システムに対する入力) がなくなるまで繰り返す．

4.3.8.2 最短経路計算

全てのイベントを処理した後，ChordAnalyer は最短経路を計算する作業に移る．これはメソッド completeGraph() によりグラフを閉じることにより自動的に行われる．

completeGraph() を呼び出された ChordAnalyzer インスタンスは，グラフを完結させるためにグラフ領域に終点ノードを追加し，最近の追加ノード集合との間にエッジを結ぶ．終点ノードについても実体は起点ノードと同様に特殊な ChordSymbol インスタンスであり，このノードと他のノードとの間の距離も例外として 0 として計算される．これらは最初及び最後のイベントに対する和音候補についても最短距離計算に含めるための処理である．

続いて，ChordAnalyzer は自身のメンバーである editable フラグをオフにする．このフラグは初期値として真であり，真の時のみ appendNodeSet などのグラフ領域への変更が許可される．

最後に，以上により得られたグラフより起点ノードから終点ノードまでの最短経路を計算する．最短経路はダイクストラ法に従って求められる．

4.3.8.3 和声進行データの出力

ChordAnalyzer は和声データの出力 (文字列化) も担当する．

toString() は得られた最短経路の各和音情報を列挙し，それを出力する．また，toGraphviz() は ChordAnalyzer の生成した和音候補グラフとその最短経路を Graphviz 形式のグラフデータで出力する．

4.3.9 AnalyzeText

クラス AnalyzeText は，main 関数を備えたプログラムの窓口部分である．入力ファイルにコードネーム列を受け取り，それから和音候補とグラフを生成して最短経路を求め，指定されたファイルへの結果の出力を行う．手順は以下の通りである．

1. GraphAnalyzer インスタンスを生成する．
2. 入力ファイルより改行区切りでコードネームを読み込み，その各コードネームに対して以下を実行する．
   (a) コードネームより和音の根音を得る．
   (b) コードネームより構成音を ChordDiscriminator を使用して特定する．
   (c) 構成音より KeyManager を使用して，考えられる調 (適合調) の集合を得る．
   (d) (a-c) により得られた根音・構成音と，適合調の各要素を使用して適合和音の ChordSymbol 集合を得る．
   (e) GraphAnalyzer の appendChordSet メソッドにより (d) で得られた集合をグラフに加える．
3. GraphAnalyzer により最適な和声進行解を計算する．
4. 3 で得られた和声進行解をテキストファイルとして出力する．また，グラフ全体を Graphviz 形式ファイルで出力する．

第5章　実験と考察

5.1　実験1: 基本的なカデンツの解析

提案システムの検証のため，まず和声学の教科書である『和声:理論と実習』[19]pp 38–39 に記載された基本的なカデンツパターン 19 種を対象として解析を行った．解析はハ長調 (**C**) における各カデンツのコードネーム列のみを与え，そこから調と音度を判断させた．与えたコードネーム列と正解となる和声進行は表 5.1 である．なお，分類は原著に従ったカデンツの種類を示す．

表 5.1: 基本的なカデンツパターン

分類	和声進行	コードネーム列	図位置
K1	I → V → I	C → G → C	図 5.1 左上
K1	I → V → VI	C → G → Am	図 5.1 右上
K1	VI → V → I	Am → G → C	図 5.1 左中
K1	I → VI → V → I	C → Am → G → C	図 5.1 右中
K2	I → IV → V → I	C → F → G → C	図 5.1 左下
K2	I → II → V → I	C → Dm → G → C	図 5.1 右下
K2	I → IV → II → V → I	C → F → Dm → G → C	図 5.2 左上
K2	I → IV → V → VI	C → F → G → Am	図 5.2 右上
K2	I → II → V → VI	C → Dm → G → Am	図 5.2 左中
K2	I → IV → II → V → VI	C → F → Dm → G → Am	図 5.2 右中
K2	VI → IV → V → I	Am → F → G → C	図 5.2 左下
K2	VI → II → V → I	Am → Dm → G → C	図 5.2 右下
K2	VI → IV → II → V → I	Am → F → Dm → G → C	図 5.3 左上
K2	I → VI → IV → V → I	C → Am → F → G → C	図 5.3 右上
K2	I → VI → II → V → I	C → Am → Dm → G → C	図 5.3 左中
K2	I → VI → IV → II → V → I	C → Am → F → Dm → G → C	図 5.3 右中
K3	I → IV → I	C → F → C	図 5.3 左下
K3	VI → IV → I	Am → F → C	図 5.4
K3	I → VI → IV → I	C → Am → F → C	図 5.3 右下

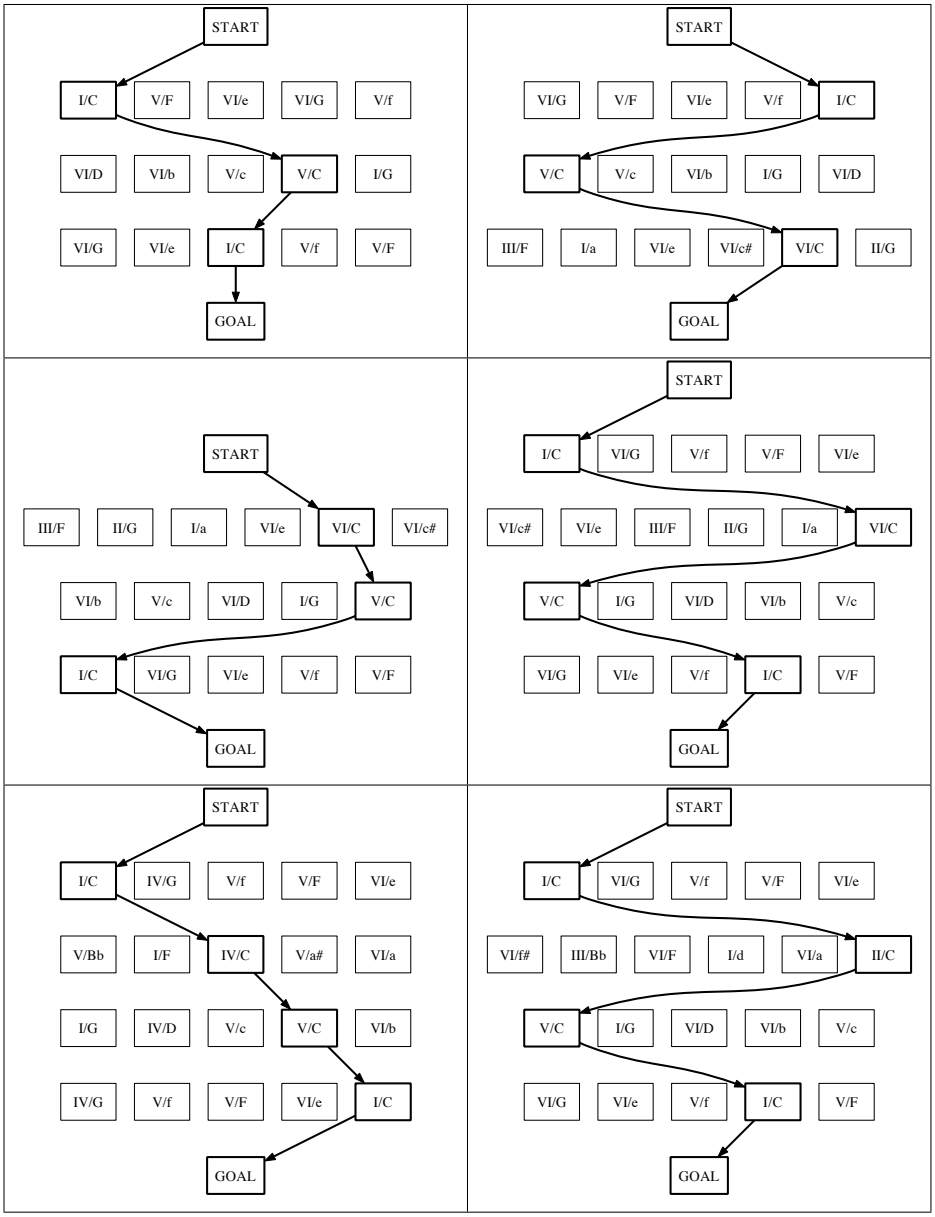

図 5.1: 基本的なカデンツの解析結果 1

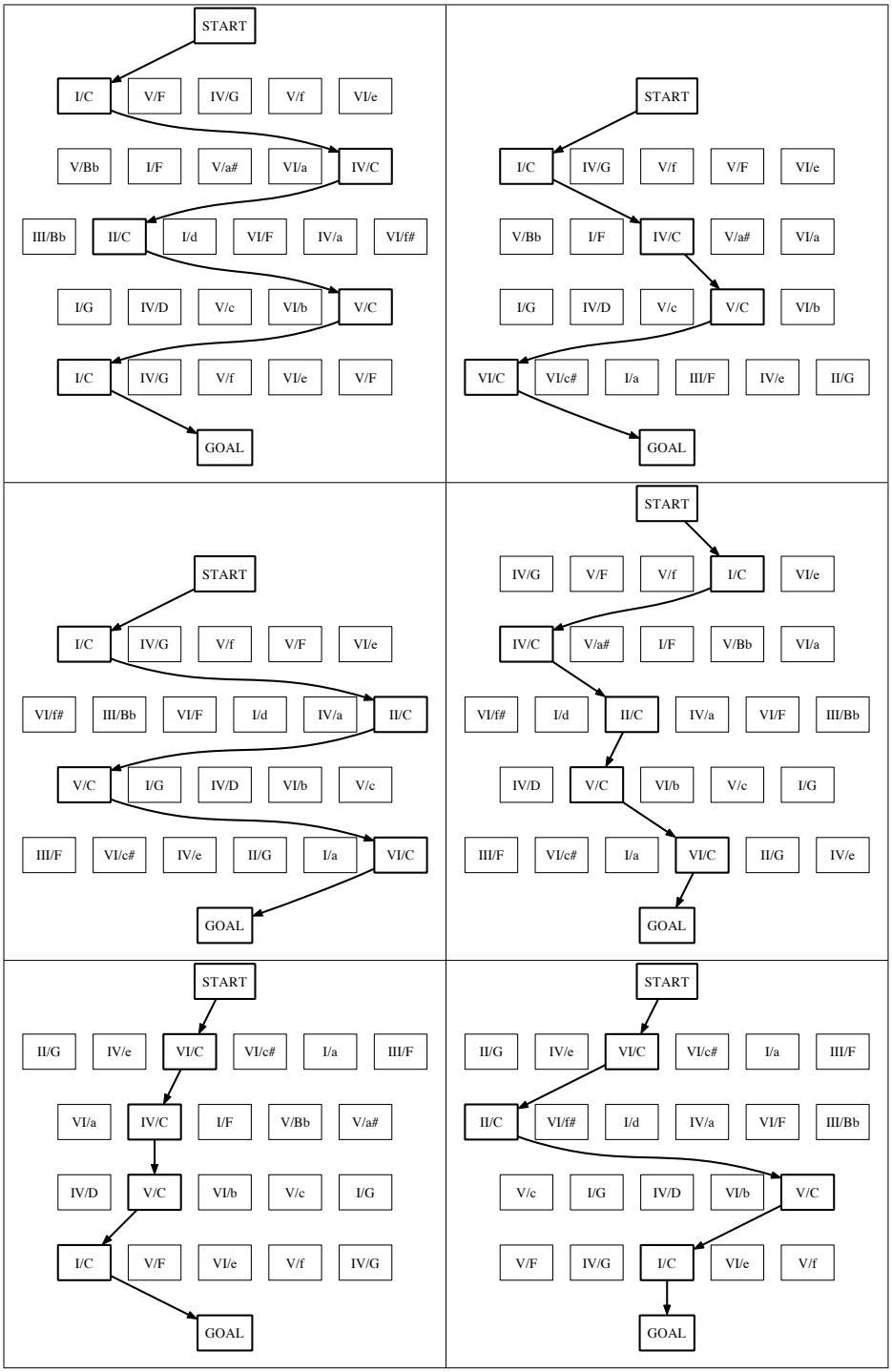

図 5.2: 基本的なカデンツの解析結果 2

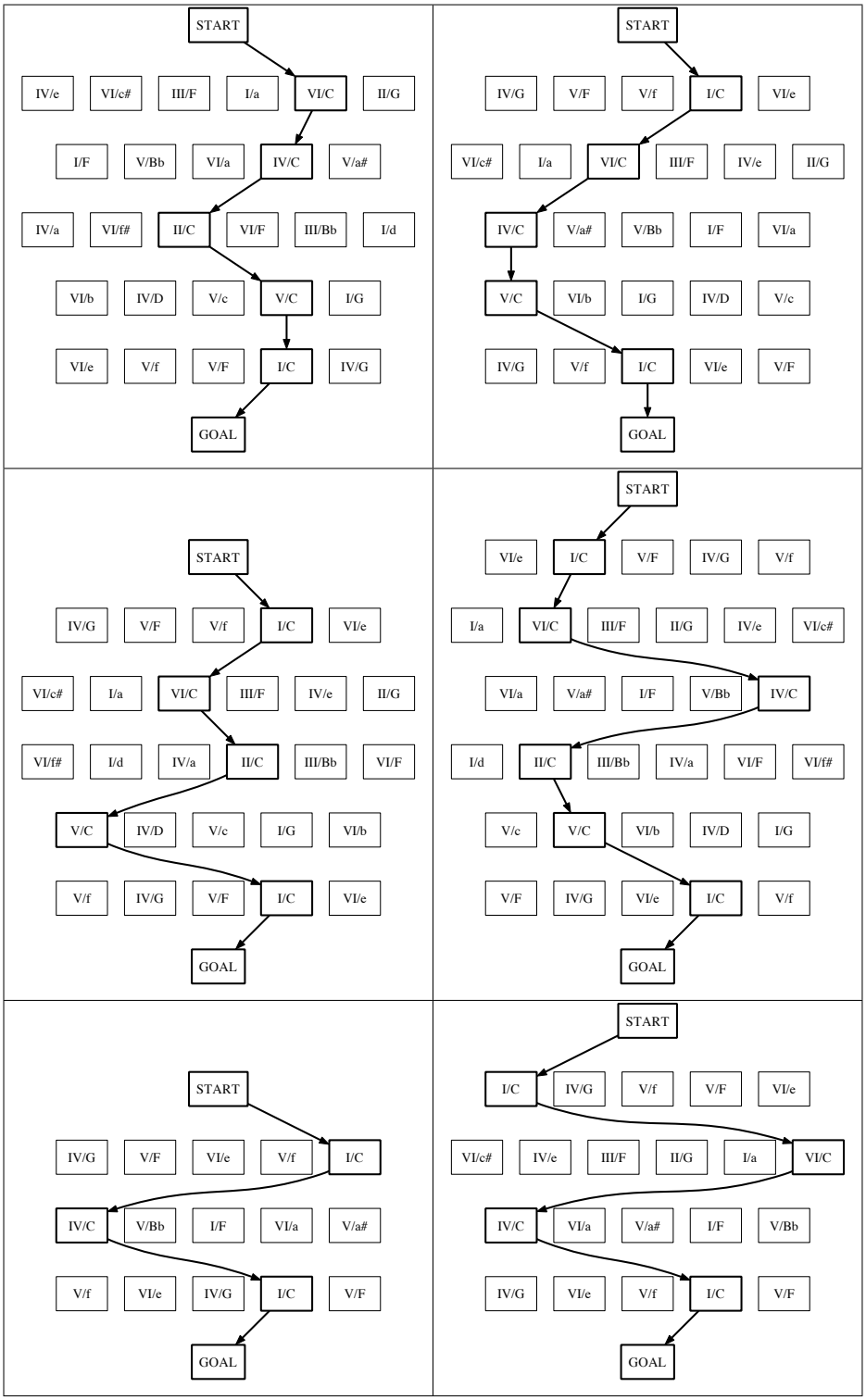

図 5.3: 基本的なカデンツの解析結果 3

実験の結果得られた和音候補グラフ及び最適経路が図 5.1・図 5.2・図 5.3・図 5.4 である．各図は和音候補グラフを示しており，四角形がノード，すなわち和音候補としてシステムが列挙したものである．ノードは階層的に配置されており，進行は一番上の起点ノード (START) から一番下の終点ノード (GOAL) に向かって下方向に時間経過がある．同じ階層に並んだノード同士は同じイベントに対する和音解釈であるといえる．ノード間を結んである線はシステムが提示した最適経路，すなわち最終的なシステムの出力となった和声進行である．実際には各階層の全てのノードと前後の階層の全てのノードがそれぞれ進行の候補としてエッジで結ばれているが，グラフ出力では見た目が煩雑になるため省略した．

グラフに示されている通り，システムは和声進行の候補としてハ長調を含んだ和音候補を各イベントから生成することができており，ほぼ全てのカデンツパターンについて提案システムは調及び音度を正しく認識した．

唯一正しく認識できなかったカデンツパターンが VI/**C** → IV/**C** → I/**C**(図 5.4) である．このパターンでは，提案手法は全体の調をヘ長調 (**F**) と誤認した．和音候補としては進行の全ての部分にハ長調 (**C**) が現れている．

5.2 実験 2: 実際の楽曲からの解析

続いて，実際の楽曲から抽出したコードネーム列を用いて解析を行い，和声進行の正解データとの一致数で精度を測った．使用したコードネーム列は Beethoven の Piano Sonate, Op49, No1 と No2 のそれぞれ 2 楽章，合計 4 楽曲のものである．

それぞれの楽曲の正解データとの一致数，正解データの和音数，一致率は表 5.2 である．一致数は提案システムの提示した和音と正解データの和音が一致した数を，総和音数は正解データ及び入力データの長さを，一致率は

$$\text{一致率} = \frac{\text{一致数}}{\text{総和音数}} \times 100 \tag{5.1}$$

で計算された正解データと出力データの一致した割合であり，精度を意味する．提案システムは入力として与えられた全てのコードネームのそれぞれに対して，必ず一つの出力を行うため，入力データ数・出力データ数・正解データ数は必ず等しくなり，一対一対応する．すなわち一致率は適合率 (Precision) であり，再現率 (Recall) でもある．また，二つの方法でその平均値を計算した．一つ目は総和音数あたりの正解率であり，楽曲の集合を M とすると

$$\text{総和音数平均} = \frac{\sum_i^M \text{一致数}_i}{\sum_i^M \text{総和音数}_i} \times 100 \tag{5.2}$$

で計算される．対して二つ目は 1 楽曲あたりの平均正解率であり

$$\text{楽曲平均} = \frac{\sum_i^M \text{一致率}_i}{\text{楽曲数}} \tag{5.3}$$

で計算される．

表 5.2: 実際の楽曲からの解析

楽曲	一致数	総和音数	一致率 (%)
Beethoven, Piano Sonate, Op49, No1-1	138	165	83.6
Beethoven, Piano Sonate, Op49, No1-2	269	317	84.8
Beethoven, Piano Sonate, Op49, No2-1	265	287	92.3
Beethoven, Piano Sonate, Op49, No2-2	149	160	93.1
合計・総和音数平均	821	920	89.2
楽曲平均			88.4

5.3 考察

5.3.1 実験 1 について

実験 1 では和声学の教科書に記載された一般的なカデンツパターン 19 種を入力として解析を行った．カデンツは楽曲における和声の単位であり，自然言語では文に相当する．これを解析することは提案システムの基礎能力を見る上で有用であると考える．結果として，19 種中 18 種について解析を行うことができたが，唯一 VI/**C** → IV/**C** → I/**C** については調を誤認した．図 5.4 のグラフをみたところ，全てのコードネームは和音候補として正しい調を含んでおり，問題は距離計算の部分にあるといえる．

入力データ	Am	→	F	→ C
想定した正解	VI/**C**		IV/**C**	I/**C**
提案システムの出力	III/**F**		I/**F**	V/**F**

上記のように，入力データはハ長調 (**C**) とヘ長調 (**F**) の両方の調に適合する．このうちどちらが選択されるかはカデンツや音の出現頻度などから判断されるのが一般的である．提案システムはカデンツ単体の認識という点では誤認したものの，この誤認は問題とならないと考える．

その理由は，実際にはこのようなカデンツパターンが単体として出現することはごく稀だからである．一般にこのようなカデンツは，前半に同じ調の他のカデンツ進行を伴った，複数のカデンツからなる複合カデンツとして出現する．

カデンツ 1	C	→	G	→	Am					
カデンツ 2					Am	→	F	→	C	
複合カデンツ	I/**C**		V/**C**		VI/**C**		IV/**C**		I/**C**	

そして，提案システムにこのような複合カデンツを入力として与えた場合には，調の誤認はおこらず，全体は正しく認識された．また，C → G → C(図 5.1 左上) なども同様に複数の調で解釈可能であるがこちらは正しく認識できている．

以上から，提案システムと TPS は概ねにおいて和声学のカデンツの原理と一致した解答を提示しているといえる．

5.3.2 実験 2 について

実験 2 では実際の 4 楽曲から作成したコードシーケンスを入力として，その和声解析を行った．提案システムの出力と正解としたデータの一致率は 83.6%~93.1%であり，和音毎平均は 89.2%，楽曲毎平均は 88.4%となった．解析の例として Beethoven, Piano Sonate, Op49, No2, 第 2 楽章 bar 9–12 に対してシステムが出力した和音候補グラフを図 5.5 に示す．

誤認のパターンとして最も多かったのは，転調位置の誤認である．これは前後の調性を正しく認識しているにもかかわらず，その調の変更位置が正解データとずれたもので，図 5.5 でもそれがみられる．

小節		9	10		11		12	13	
入力データ	···	G	B	Em	Em	Bm	Am	G	···
正解データ		I/**G**	V/**e**	I/**e**	VI/**G**	III/**G**	II/**G**	I/**G**	
出力データ		I/**G**	V/**e**	I/**e**	I/**e**	III/**G**	II/**G**	I/**G**	

これは，正解データと異なっているものの和声進行としては誤りとは言えないと考える．誤認部分の 11 小節第 1 和音は **G** としても **e** としても解釈可能であり，両方の調性を備えた和音であると言えるからである．しかしながら，一般的にはこのような場合，小節の代わり目など拍節的な重要度が高い部分を転調と考える．提案システムでは，コードシーケンスのみから解析を行い，どの部分が小節の変わり目であり，拍節的に重要であるかは考慮しなかった．提案システムの改良案として，これらの拍節情報を取り入れることで，より人間の認識に近い調性判断ができるのではないかと考える．

続いて問題となったのは，必要以上に小さな転調を考慮してしまう問題である．これは出現頻度こそは少なかったが一つの和音を間違えたために連鎖的に誤認が続くケースが多かった．この問題は，一部の経路で転調を含んだ進行と転調を含まない進行が和音間距離の総和が等しいか転調を含んだ進行の方が小さくなっていることに起因する．このような場合により転調回数が少ない進行を優先することでこの問題は解決できるのではないかと考える．これは，転調を含んだ和音間距離に重み付けすることで実現できるが，逆に並行調の移動などを阻害してしまう可能性もあり，重みパラメータの設定には考慮が必要となる．

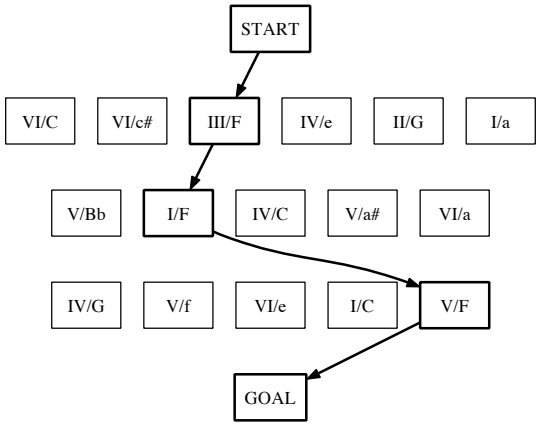

図 5.4: 基本的なカデンツの解析結果 4

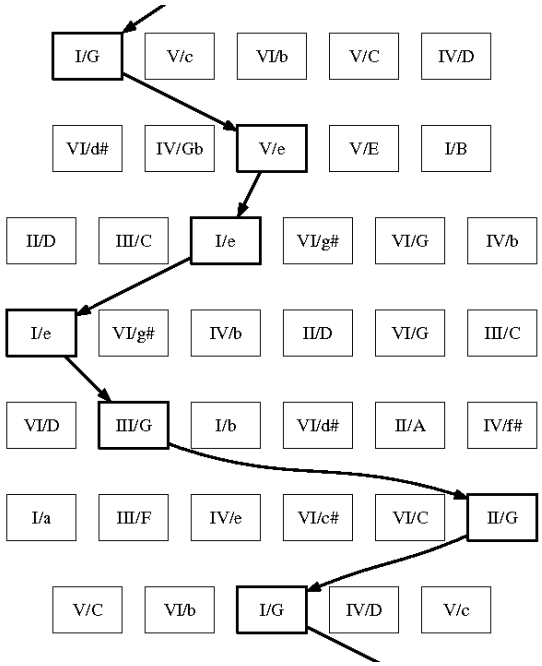

図 5.5: Beethoven, Piano Sonate, Op49, No2, 第 2 楽章 bar 9–12

5.3.3 実験のまとめ

提案手法は，Lerdahl の和音間距離とダイクストラ法を使用し，入力としてコードネームのみから和声解析を行うという点でシンプルなシステムであるが，カデンツパターンのほとんどを認識でき，実際の楽曲に対する解析も 80%以上の精度を導いた．システムとしては依然として改良の余地が残っており拍節情報の取り入れや，転調に対するペナルティを科すなどの方法でより人間の認知に近い和声解析ができるのではないかと考える．

5.4 他研究との比較

本研究と類似したものとして Rocher ら [14] のシステムが挙げられる (付録参照)．Rocher らのシステムは本提案システムと同様に TPS の距離を用いた和声解析を行うが，以下の点において本研究と異なっている．

- 本研究はテキストデータからの解析を行うのに対して，Rocher らは MIDI ファイルからの解析を行っている．Rocher らのシステムは MIDI ファイルから和音区間及び和音名の解析を行いコードネームを検知しているという点において，本研究より包括的なシステムである．本研究では和音区間と和音名の解析についてはその範囲外としたが，これは区間及び名称の解析方法については既存手法の組み合わせにより行うことができ，問題ではないと考えたからである．Rocher らも既存手法の組み合わせによりこれを実現している．

- グラフからの最適経路の探索について別のアルゴリズムを用いている．本研究ではダイクストラ法による単純な探索を行ったが，Rocher らはそれに加えて経路の足切りを試みている．結果としては同等であるが，Rocher らの手法の方が計算速度の面で勝ると考える．

- TPS による距離計算について Rocher らは初版の和音間距離ルールのみに基づいた簡易な方法を採用している．先述のように，初版の和音間距離ルールでは和声学的な見解と乖離が生じる場面が存在する．この点については本研究が優れていると言えるが，これがどの程度の精度の向上をもたらすのかは検証の余地がある．実際のところ，実験で用いた四つの楽譜データではこれが問題となる遠隔調の転調は存在しなかった．

以上は，Rocher らが実際のデータへの適応を主目的としているのに対して，本研究が部分問題として和音解析の解決と TPS の有効性検証を主目的としていることから生じるものではないかと考える．Rocher らは MIDI アレンジソフトの出力を用いて，より大規模な実験を行っているが，この結果は 83.3%と本研究に近い．

第6章　おわりに

本研究では，大きく分けて以下の二つのことをおこなった．

第一は，TPS の和音間距離についての整備である．TPS は既知の音楽理論を組み合わせにより和音間・調間の定量化をおこなうという点で興味深い理論であるが，その実装例は少なく，特に調間距離・修正和音間距離まで実装した研究は無かった．本研究ではこれらの要素の実装をおこなうために TPS の各ルールについて整備し，実装が容易な形へと原意をできるかぎり保持する形での修正をおこなった．

第二に，TPS についての実装を元に和声解析システムを提案し，その評価実験をおこなった．和声解析システムは楽曲のコードシーケンスから適合する和音と適合する調を列挙し，それらから考えられる和声進行解釈全体を示す和音候補グラフを構築し，TPS の和音間距離をもちいて和音候補間の距離を定量化することにより最短経路をもとめ，それをコードシーケンスに対する和声進行解として出力した．評価実験の結果，システムはほぼ全てのカデンツを認識することができるなど，和声学との一致を示し，実際の楽曲をもちいた和声解析でも 80%以上の精度をだした．これにより TPS の有用性について示すことができたと考える．

今後の課題としては，まずは提案システムの改良があげられる．提案システムでは，コードシーケンスのみを入力としてとっていたが，拍節構造などを参照することでより人間の認知に近い和声解析がおこなえると考える．また，より転調の少ない進行を優先するなどの手法も考えられる．これらの改良のためには，和音候補間の距離として TPS の和音間距離を直接とるのではなく，それに拍節構造など他の要素を加味した独自の和音間距離を考慮しなければならないが，これは各要素のバランスに配慮しなければおこなわなければならない．さらに，速度面での改良も考えられる．提案手法ではコードシーケンス全体を一つのグラフとして最短経路を求めたが，これは一つの楽曲につき数分を要した．リアルタイムなシステムに組み込むことを考えると，グラフを分割するなどの手法により高速化を図ることが好ましい．

もう一つの課題は，MusicXML などの実際の楽譜データからの和声解析である．このためには，楽譜から適切な和音区間を認識し，その区間内の非和声音を取り除き，分散和音なども考慮してコードシーケンスを作成するシステムが必要である．この問題を考えたとき TPS の優位な点は，和音間の距離の定量化はすなわち和声進行全体の定量化でもあるという点である．すなわち，和声進行の和音間距離総和をその和声進行に対する評価値とすることで，和声進行の定量化がおこなえる．これは他の和声解析手法にない特徴であり，他の手法に比べて応用の幅が広がるのではないかと考える．従来の研究では，コード

ネームの特定は和音区間の解析を，和声解析はコードネーム (構成音) の特定をそれぞれ必要としており，これらはウォーターフロー方式でおこなうしかなかった．しかし，和声進行を定量化することで，今度は和声解析の結果を前 2 者の解析にフィードバックできるのではないかと考える．例えば，遺伝的アルゴリズムにより和音区間を仮定し，それに対して和声進行の距離を評価値にして学習させることが可能となる．また，GTTM によって得られた拍節構造などを利用することも考えられる．

参考文献

[1] 調 - Wikipedia(日本語版). http://ja.wikipedia.org/wiki/調. 2010 年 1 月 31 日アクセス.

[2] David J. Benson. *Music: A Mathematical Offering*. Cambridge University Press, 2007.

[3] Erich Gamma, Ralph Johnson, Richard Helm, John Vlissides, et al. **オブジェクト指向における再利用のためのデザインパターン**. ソフトバンククリエイティブ, 1999.

[4] Michael Good. MusicXML: An internet-friendly format for sheet music. In *the XML 2001 Conference*, 2001.

[5] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. FATTA: Full automatic time-span tree analyzer. In *the 2007 International Computer Music conference*, 2007.

[6] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. Melody expectation method based on GTTM and TPS. In *the 9th International Conference on Music Information Retrievalconference (ISMIR)*, pages 107–112, 2008.

[7] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. Melody morphing method based on GTTM. In *the 2008 International Computer Music conference*, 2008.

[8] Carol L. Krumhansl. *Cognitive Foundations of Musical Pitch*. Number 17 in Oxford Psychology Series. Oxford University Press, 1990.

[9] Fred Lerdahl. *Tonal Pitch Space*. Oxford University Press, 2001.

[10] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, 1983.

[11] H. C. Longuet-Higgins and M. J. Steedman. On interpreting bach. *Machine intelligence*, 6:221–241, 1971.

[12] J. F. Paiement, D. Eck, and S. Bengio. A probabilistic model for progressions. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 312–219, 2005.

[13] Christophe Rhodes, David Lewis, and Daniel M"ullensiefen. Bayesian model selection for harmonic labelling. In *Program and Summaries of the First Internatinal Confernece of the Society for Mathematics and Computation in Music*, pages 18–20, 2007.

[14] Thomas Rocher, Matthias Robine, Pierre Hanna, and Robert Strandh. Dymanic chord analysis for symbolic music. In *Proceedings of the International Computer Music Conference (ICMC 2009)*, 2009.

[15] A. Sheh and D. P. W. Ellis. Chord segmentation and recognition using EM-trained hidden markov models. In *Proceedings of the 4th International Conferrence on Music information Retrieval (ISMIR)*, pages 183–189, 2003.

[16] D. Temperley. *The Cognition of Basic Musical Structures*. The MIT Press, 1999.

[17] 小方 厚. **音律と音階の科学**. 講談社, 2007.

[18] 西田 昌史, 東条 敏, and 佐藤 健. HPSG を用いた楽曲の和声解析. **情報処理学会研究報告**, 2002-MUS-47:127–134, 2002.

[19] 島岡 譲 et al. **和声*:* 理論と実習**. 音楽之友社, 1964.

[20] 長嶋 洋一, 橋本 周司, 平賀 譲, and 平田 圭二. *bit* **別冊 コンピュータと音楽の世界*:* 基礎からフロンティアまで**. 共立出版, 1998.

謝辞

本研究を進めるにあたり，熱心な御指導・御助言をいただきました本学 東条敏教授に深く感謝いたします.

また，御指導をいただきました，本学 島津明教授，飯田弘之教授，鶴岡慶雅准教授，情報通信研究機構 鳥澤健太郎様 (元本学准教授) に感謝致します.

最後になりますが，助言・協力をいただきました，萩原信吾氏，的場隆一氏，池田将之氏をはじめとする東条研究室の皆様に厚く御礼申し上げます.

付録 A　記号譜面のための動的和音解析

Thomas Rocher, Matthias Robine, Pierre Hanna, Robert Strandh
LaBRI
University of Bordeaux 1
351 avenue de la Libration F 33405 Talence, France
simbals@labri.fr

概要

本稿では，我々は記号楽譜 (Symbolic Music) からの和声認識を行う新しい手法を提案する．この手法では，全ての適合する和音からなるグラフを構築し，そのグラフにおける最適なパスを選択する．ノートの集まりから互換性のある調と和音を考慮することで和音の候補を列挙するために，ルールベースの方法を導入した．異なる和音候補間のコスト計算には，Lerdahl の提案する距離を用いている．また，和音候補内での最適な経路の選択には動的計画法が使用されている．ジャンルごとの MIDI 楽曲データベースにより実験を行い，提案手法と Temperley の *Melisma Music Analyzer* との比較を試みた．その結果，提案手法はより効果的であり，和音のルートだけでなく，そのモード (長調か短調か) についても解析することが可能であった．本提案手法はオープンであり，さらなる種類の和音に対しても，それらを扱う的確なルールを記述すれば対応可能である．

A.1　導入

調や和音のような和声的パラメータは，西洋音楽における本質的な要素である．なぜなら調は旋律と和声が組み立てられるための調性の中心として解釈可能であり，コード進行は楽曲の和声的構造を構成するからである．そのため，和音の認識により，楽曲やその部分から大まかな和声進行を把握することができる．しかし，そのような課題は複雑である．なぜならば，コードは非一様なパターンに変形しているだろうからである．さらに困難なことに，曲の全てのノートが和声的なわけではない．その一部は装飾音かもしれず，よって進行中のコードに属していないことがありえる．和音認識の応用範囲は幅広く，自動的にコードに対してラベル付けを行う楽曲編集ソフトウェアや，背景のコードを検知して演奏されたノートに対して適正な三度音，四度音，五度音を発音するハーモナイザなど

に組み込むことが考えられる．楽曲情報検索 (MIR) もまた，コード進行に基づいた楽曲類似度の計算により，本研究分野の恩恵を受けるだろう [2].

和音認識についての研究の大多数は，学習段階を取り入れ，隠れマルコフモデル (HMM) を使用し，特に音声を扱ったものである．Sheh と Elis は 12 次元クロマベクトル特徴と期待値最大化法 (EM 法) を学習段階で使用した [15]．Bello と Pickens は同様の手法を採用し [8]，五度圏 (図 A.5) を考慮した調性距離によって成り立つ遷移行列を導入することによって，モデルにいくつかの音楽情報を加えた．Lee らは調性セントロイドという異なる特徴を提案し，学習段階における EM アルゴリズムを不要とした [9]．Cabral らは，学習データベースに応じて適正な学習プロセスと適切な特徴を自動的に構成するツールを使用した [5]．これらの技術は [12] において詳解・比較されている．これら全ての研究において，与えられた特徴 (一般にクロマベクトル) に従って和音の確率を定義するために学習段階が必要である．また，これらは皆，均一時間分割を用いている．

本稿では，我々はノートのリストとして表現されるような，記号的楽譜に焦点を置く．記号的和音検知の論文においても，Paiement ら [11] や Rhodes ら [14] のようないくつかの統計的手法では学習が用いられている．このような手法の結果は訓練に用いたデータベースに依存するため，我々は別の手法をとることを決定した．残りのシステムはルールベースのものである．ルールは一般的に音楽理論由来のものであり，実装には動的計画法が用いられることが多い．Temperley が主張するところとして，動的計画法は「左から右に」の選考規則システムを実現するのに誠に効果的な方法であり，システムは今までのところ音楽をリアルタイムで聴く過程と類似していて聴いた全てのものの好ましい解析が可能である．1999 年，Sleator と Temperley は *Melisma Music Analyzer* を提案した [16]. *Melisma* は，与えられた音楽入力よりまず拍子の解析プロセスを適用し，続いて根音の推定を行い，それに基づいて動的計画法により，それぞれの和音を検知する．より近年では，Illescas らもまた検知した和音の機能推定に動的計画法を用いている [7]．このシステムは正しいピッチのスペリングと拍子情報を要求する．

ここで我々は，記号的で多声の入力より非一様の和音列を推定する新しい手法を提案する．*Melisma* において，それぞれの検出された和音は根音とともにラベル付けされる．本提案手法では，それに加えてモード (長調か短調か) もまた提供する．この手法では適合する和音群より配列を選択するのを動的計画法に頼っている．適合する調についても，それぞれの調によって定まる二つの和音の間の遷移コストにより考慮される．拍子情報やピッチのスペリングは必要とはしない．和音の表象についての議論が第 A.2 節で提案された後，第 A.3 節では提案手法について触れる．続いて，第 A.4 節では実験とその結果について述べ，最後に第 A.5 節では本研究により広がる将来の課題についてまとめる．

A.2　和音の表象

和音は異なる複数の方法により定義可能であり，単純には，同時に鳴るいくつかのノートの知覚であると定義される．しかしながら，この第一の定義はそれぞれの小節に対して

一つの和音でラベル付けされるであろうジャズのコード表記と完全に一致するわけではない．このような場合，例えばピアニストは特定の和音の全てのノートを演奏しないはずである: 彼に課される唯一の制約は特定の和音に**大部分が**属するようなノートの集まりを演奏することだ (和音に属さないようないくらかの装飾音も演奏されるだろう)．これら二つの可能な定義の違いは図 A.1 によって例証される．採用された定義に依存して，この引用譜は 8 つの和音とも，一つの和音とも見ることができる．

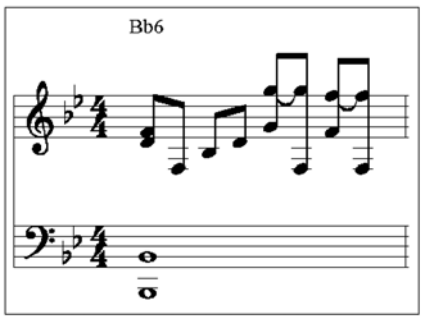

図 A.1: クイーンのボヘミアン・ラプソディからの引用譜．第一の定義に従うなら，この小節には 8 つの異なる和音がある (同時に鳴っているノートの集まりの数ある)．第二の定義に従うなら，この小節には一つの和音しかない (和音ラベル Bb^6 に象徴される)．

本稿において，我々は後者の定義を選択し，**和音**(*chord*) という単語をノートの並びや同時に鳴るノートの集まりに一般的なラベルという意味で (図 A.1 における Bb^6 和音ラベルのように) 使用する．同じ和音に含まれるこれらの連続した和音や和音の集まりは，その和音の**ノート・セグメント**と呼ばれる．与えられた和音のそれぞれのノート・セグメントはその和音の全てのノートを含んでいるとは限らず，また，装飾音 (その和音に属さない音) を含んでいるかもしれない．例えば，(C,E,Eb) と (C,G) はともに和音 C^7 のコード・セグメントである．本稿では，一つの和音は以下の三つのパラメータをもっている．

- 根音 (ルート．和音が作られる土台となるノート)
- 型 (根音からの相対度による和音の構成音程)
- モード (長調，短調，或いは未定義)

A.3 動的和音解析

本節では，まずノート・セグメントを用いた新しい時間分割手法を提案する．続いて，和音候補のグラフが生成され，動的処理によりグラフからの最適経路の選択が行われる．

A.3.1　時間分割

和音検知処理における第一の課題は適切な時間分割を探し出すことである．音声をベースとした手法 [5, 9, 15] では，この目的のためにタイム・フレーム (time frames) を用いる．解析枠の長さを設定するための時間単位として MIDI ticks(つまりミリ秒時間) を用いることで，記号的楽譜においても類似した方法をとることができる．しかし我々は異なる手法をとり，新しい和音は新しいインスタンスと同時にしか発生しないと見なすこととした．すなわち，少なくとも一つ以上の新しいノートが演奏開始されるか，演奏中のものが停止されるときである．従って我々は，同時に鳴る全てのノートが，同じように同時に開始，終了するように，[6] で導入されるようなホモリズミックな変形処置を取り入れる．それ故にノート間でのオーバーラップは発生しない．この時間分割は異なる複数のタイム・セグメントを定義する．それら各々は，同時に開始され同時に終了するノートの集まりにより形成され，新しい和音の開始位置となる可能性をもっている．この変形作業が演奏され聴かれる音楽の様式には影響を与えないという点は重要である．このホモリズミック変形は図 A.2 のように説明できる．この変形が完了した後，それぞれのノート・セグメントに対して和音候補の列挙を行う．

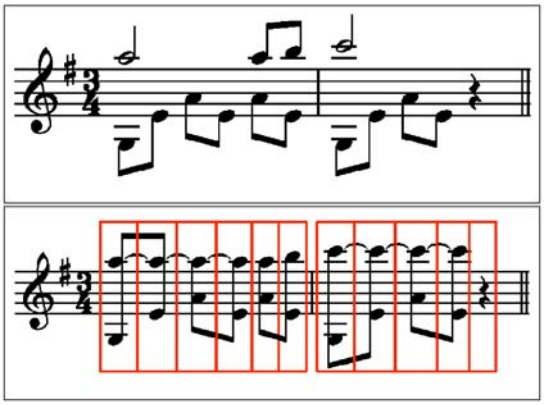

図 A.2: 上部: ある楽譜の一部．下部: 同じ楽譜のホモリズミックな変形．ノート・セグメントは赤線部分で示されている．

A.3.2　和音候補グラフ

ノート・セグメントは動的処理の観測 (observation) の構成要素となる．それぞれの観測から仮説のリストが作られ，それらの仮説は和音候補となっている．本提案手法において，和音候補は適合する和音と適合する調からなる対によって成り立っている．ルールベースのアルゴリズムがそれぞれのノート・セグメントと適合する和音と調を決定するために用いられる．その次に，和音候補のグラフが組み立てられる．あるセグメントのそれ

ぞれの和音候補は次のセグメントの全ての候補とリンクされ，それにより有向非巡回グラフが形成される．

A.3.2.1 適合調

本提案手法はそれぞれのノート・セグメントと適合する調を，ルールベースのアプローチを用いることより列挙する．この課題には複数の異なるルールを用いることができるだろう．本稿で我々は，もしノート・セグメントのそれぞれのノートがある調の構成音程である場合，その調は適合であるという定義を使用した．すなわち，セグメントのそれぞれのノートはその調のスケールに属していなければならない (短調に対しては旋律的短音階を用いる)．例えば，ノート・セグメントが (C,E) ならば，適合する調は C_{Maj}，F_{Maj}，G_{Maj}，A_{min}，そして E_{min} である．なぜならば，C と E は共にこれらの調のスケールに属するからである．

A.3.2.2 適合和音

適合調と同じように，和音がノート・セグメントと適合であるかの決定についても複数のルールが定義されているだろう．我々は次のようなルールを用いて，ノート・セグメントと和音が適合であるための条件を和音の型によって複数のことなるものを定義した．

- 長/短 三和音: このような和音は，考慮するセグメントのそれぞれのノートがその和音に属するときに適合である．例えば，ノート・セグメント (C,E) は二つの適合する三和音 A_{min} と C_{Maj} をもつ．
- 長/短 7th コード: このような和音は，考慮するセグメントのそれぞれのノートがその和音に属し，且つ根音と七度音が存在するときに適合である．このようなルールは，例えば (E,G,B) や (E,G) のようなノート・セグメントが C^7_{Maj} と適合するのを防ぐためである．
- 長/短 9th コード: このような和音は，考慮するセグメントのそれぞれのノートがその和音に属し，且つ根音，五度音と九度音が存在するときに適合である．
- 長/短 11th コード: このような和音は，考慮するセグメントのそれぞれのノートがその和音に属し，且つ根音，五度音と十一度音が存在するときに適合である．
- それ以外の和音についても，${}^7_{min}5\flat$ 和音のような場合に適合でありえる．これらの和音が適合であるためにはそれぞれのノートが必須である (例えば，${}^7_{min}5\flat$ 和音は考慮するセグメントが根音，短三度音，変五度音と短七度音を含む場合に適合である)．

提案システムは将来的により多くの和音型を許容できるように修正ができるようになっている．その場合には，与えられたノート・セグメントがなにかしらの新しい和音と適合するための新しいルールを記述しなければならない．

A.3.3 和音候補列挙

最終的に列挙される和音候補は，適合する調と適合する和音の全ての可能な組み合わせである．n 個の調と m 個の和音が適合ならば，$n \times m$ 通りのペアが列挙される．例えば，適合する和音として C_{Maj} と A_{min}，適合する和音として C_{Maj} と G_{Maj} があるとき，和音候補として (C_{Maj},C_{Maj}), (G_{Maj},G_{Maj}), (A_{min},C_{Maj}), (A_{min},G_{Maj}) が列挙される．与えられたセグメントにたいして適合する和音がない場合，一つ前のコード・セグメントのその適合調と組み合わせられた和音候補を仮定することを我々は選択した．適合する和音がない場合の調検知についての説明もこれと同様である．適合する調を組み立てることができない場合，一つ前のノート・セグメントのそれを置くことを仮定した．これらの仮定は，二つの連続したノート・セグメントが高確率で同じ和音の部分であることから理にかなっているといえるだろう．最後に，和音の出力は根音とモード (長調か短調か) のみである．つまり，我々のシステムによってサポートされる和音の型であったとしても，考慮されるのは対応するモードのみとなる．例えば，和音が C^7 として検知されたとしても，システムによって C_{Maj} として扱われる．このように我々は現在のところ根音とモードに焦点を合わせており，和音型の評価は今後の課題の一つである．

A.3.3.1 和音遷移コスト

二つの連続したノート・セグメントについて和音候補が列挙されたら，一つ目のセグメントのそれぞれの和音候補から次のセグメントのそれぞれ和音候補に，エッジが結ばれる．このエッジは，二つの和音候補間の遷移コストによって重み付けられる．この遷移コストは，異なる適合和音と異なる適合調の両方によって説明されなければならない．

そこで我々は，遷移コストとして Lerdhal の距離 [10] を用いることを選んだ．この距離はベーシック・スペースという概念に基づいている．Lerdahl はある調においてのある和音のベーシック・スペースを次ような幾何学的な重ね合わせによって定義した．

1. 与えられた調の半音階のピッチ (chromatic level)
2. 与えられた調の全音階のピッチ (diatonic level)
3. 与えられた和音の三和音のピッチ (triadic level)
4. 与えられた和音の根音と五度音 (fifth level)
5. 与えられた和音の根音 (root level)

図 A.3 は，調 C_{Maj} における和音 C_{Maj} のベーシック・スペースを表したものである．調 K_x の和音 C_x を (C_x, K_x) と表すとしたとき，$x = (C_x, K_x)$ から $y = (C_y, K_y)$ の遷移コストは Lerdahl によって次のように定義される:

$$\delta(x \to y) = i + j + k$$

ここで i は五度圏 (図 A.5) における K_x と K_y の間の距離，j は五度圏における C_x と C_y の間の距離，k は x のベーシック・スペースと y のベーシック・スペースを比較したときの共通しないピッチ・クラスの数である．

したがって，距離は 0 から 13 の整数コストを備えており，適合和音と適合調の両方がコスト計算に関係するという理由から，本提案手法の遷移コストに非常に適切である．$x = (C_{Maj}, C_{Maj})$ から $y = (G_{Maj}, G_{Maj})$ の和音遷移の計算は図 A.4 のようになる．ここで，五度圏において C_{Maj} から G_{Maj} への移動は 1 ステップであるため，$i = j = 1$ となる．$k = 5$ は y のベーシック・スペースと x のベーシック・スペースの比較によって異なっているピッチの数である (図中の下線部)．よって距離は $1 + 1 + 5 = 7$ となる．

A.3.3.2　動的処理

全ての和音候補間のグラフが形成されたら，その最適経路を探索しなければならない．この課題は，動的計画法 (dynamic programming)[1] にの範疇である．グラフの左から右へ，それぞれの和音候補へのただ一つのエッジが保持される．このエッジを選択する方法をいくつか考慮したが，実験の結果，図 A.6.c のように，それぞれの候補へのコストを最小限にするようなエッジを保持する方法を我々は選択した．候補の経路全体のコスト総和を最小限にするようなエッジを保持する方法など，別の可能性の探求は今後の課題である．

最終的な経路の数は，最後のノート・セグメントに対する和音候補の数となる．そして，プログラムはそのコスト総和が最小であるような経路を出力する．

A.3.3.3　全体の計算

全体の流れは図 A.6 によって図示される．グラフの構築は二つのステップからなっている．まず，それぞれのノート・セグメントに対する和音候補が決定される (図 A.6.a)．続いて，前者のノート・セグメントのそれぞれの和音候補から後者のそれぞれの和音候補の間のコストが計算される (図 A.6.b)．その後に，動的処理が以下を行う: 与えられた和音へのエッジを一つだけ保持する (図 A.6.c)．最後に，総コスト和が最小となる経路が選択される (図 A.6.d)．

引用譜に対する全体の処理計算の例は図 A.7 である．それぞれのノート・セグメントについて和音が計算され，連続する同じ識別和音をもつセグメントは，各々の和音の境界を形成するためにマージされる．この例では，最初の二拍には Eb_{Maj}，最後の二拍には Ab_{Maj} の和音が検知された．

A.4　実験

提案手法の評価を行うために，我々は和音の特定された正解データ付きの MIDI 曲集を集め，*Melisma Music Analyzer* との精度比較を行った．

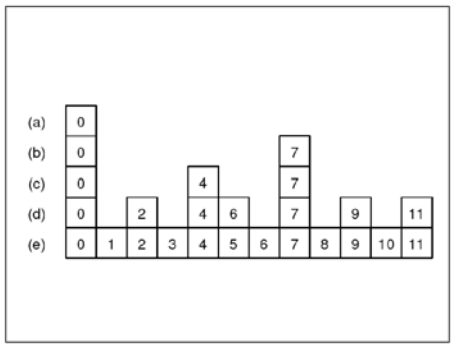

図 A.3: 調 C_{Maj} における和音 C_{Maj} のベーシックスペース．Level (a) から (e) は半音階，全音階，三和音，五度音，根音のレベルに対応する．

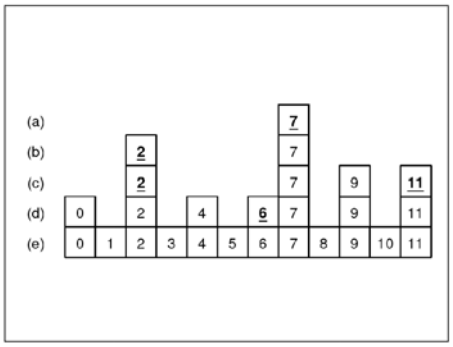

図 A.4: $\delta((C_{maj}, C_{Maj}) \rightarrow (G_{maj}, G_{maj})) = i + j + k = 1 + 1 + 7$. 下線は共通しないピッチ．

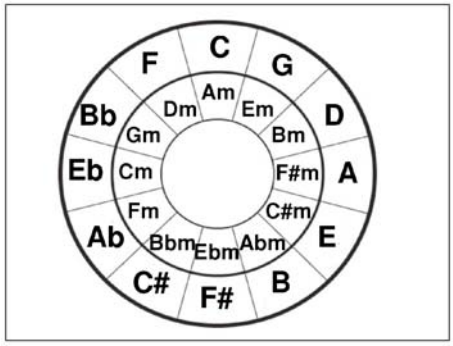

図 A.5: 五度圏

A.4.1 データベース

選ばれた 155 個の MIDI ファイルは，ソフトウェア Band-In-A-Box[1]により出力されたものある．Band-In-A-Box は MIDI 楽曲のアレンジソフトであり，西洋音楽で用いられるコード進行に対して伴奏を生成する．使用した Band-In-A-Box ファイルは，Band-In-A-Box Independent Users Group File Archives 2[3] からとった伴奏である．これらのファイルを用いることで，正解データの (Band-In-A-Box により特定された) 和音にアクセスすることができ，和音列に対応した MIDI 伴奏を生成することができる．楽曲はアーカイブの中で既に，ラテン，ポップ，クラシック，ジャズの四つの異なるスタイルに分類されている．データベースの詳細は表 A.1 である．

	楽曲	合計 MIDI Tick
ラテン	37	1177
クラシック	8	266
ポップ	45	1866
ジャズ	65	2405

表 A.1: 実験に用いた MIDI データベース

A.4.2 評価手順

しばしば音声からの和音認識手法はフレーム単位で評価される [8, 9, 15]．それぞれのフレームについて，和音が推定され，全体の精度は全フレーム中の正解データ (ground truth) と一致するフレームの割合である．同様に，データベース中の各楽曲について，分解能 (tick) 毎に推定された和音と正解の比較を行うことで，拍単位の評価を行った．我々は Melisma と提案手法の両方について，このデータベースでの精度をテストした．

A.4.2.1 Melisma の評価

Melisma Music Analyzer は Sleator と Temperley によって開発されたソフトである [16]. Temperley は J.S. バッハの**平均律クラヴィーア曲集**の 48 のフーガと抜粋の全集を使用してこのプログラムの根音抽出精度を検証した．このデータベースにおける精度は 83.7%であった．

Melisma は検知した各和音の根音のみを提示するため，与えられた楽曲に対する精度は，全拍中の推定された根音が正解データと一致している拍の割合で計算される．各音楽ジャンルごとの全体的な精度を算出するために，以下の二つの異なった平均が求められた．

- tick 平均，曲の精度が MIDI tick(分解能) 数によって重み付けられる

[1] `http://www.pgmusic.com/products_bb/htm`

- 曲平均，各曲の精度は同じ重みをもつ

すなわち，全体の精度は，データベースの全ての曲に対して二つの平均を求めることによって計算される．

A.4.2.2 システムの評価

我々のシステムは根音だけでなくモード (長調か短調か) についても推測するため，二つの異なる評価を行うこととした．第一の評価方法は *Melisma* のものと同様である: 根音のみが取り出され，正解データと比較される．第二の評価方法は以下のルールに従って根音とモード (長調か短調か) の両方の精度を算定するものである: 正解データで示された和音がモードを持っているならば，それは推測された和音のモードと比較される; しかし，正解データで示された和音のモードが未定義ならば (*dim* 及び *sus* のような場合)，根音のみが Melisa の評価と同様に比較される．これらの評価を行うことによって，Melisma の精度との直接の比較が可能になる．

また，システム精度の最適化を考慮に入れて，どの和音型 (トライアド，7th，9th，11th，13th) を決定するために，我々はさらにいくつかの評価を行った．

A.4.3 結果

この節では，まず複数の異なる和音型を扱ったシステムの結果を示す．次に，根音推測における提案手法と *Melisma* の結果の比較を提示する．最後に，検知された和音の根音とモードを評価した結果について述べる．

A.4.3.1 サポートされる和音型の最適化

サポートする和音の数を増やすことは，各ノート・セグメントに対して考えられる和音候補を増加させる．それにより，二つの連続したセグメントが二つ和音候補が同じになる確率も増加する．同様に，考慮される二つのノート・セグメントの間で和音の変更があったとしても，同じ和音候補を保持するようにシステムが選択する可能性が生じる (図 A.8)．従って，サポートする和音型の数を増やすことと，それによって増加するエラーについて均衡をとる必要がある．

サポートする和音型の最適化についての探索は，表 A.2 で示されるの結果より，11th コードまでがよいように思われる．7th コードを取り扱うことは全体の精度の上で特に効果的である点を指摘しておく．ラテンやジャズにおいては特に影響が大きい (この二つのジャンルにおけるスコアは相対的に 19%と 16%増加した)．この主たる理由は，この二つのジャンルが三和音より複雑な和音を多く含むことである．9th コードを扱うことは先の二つのジャンルについてのみ真に有益である (これらのスコアはさらに 4%増加した)．11th コードの使用は，全てのスコアについて非常に小さな影響しか与えなかった．最後に，

13th コードを扱った場合，全体的にスコアは著しく低下した．特に，その傾向はこれもまたラテンとジャズで顕著である (これらのスコアは相対的に 8%と 5%低下することとなった)．これは単純な例を示すことによって説明できる: (A, C, E, G) のようなコード・セグメントを考えると，三和音，7th コード，9th コード，11th コードをサポートした場合にのみ A_{min} は適合である．しかし 13th コードを考慮すると，七度音，九度音，十一度音のない C^{13}_{Maj} として認識できてしまう．それにより，13th コードを扱ったとき，(A, C, E, G) は適合和音として A_{min} と C^{13}_{Maj} の両方を持ってしまい，図 A.8 であげたような間違いを導いてしまう．以上がシステムでサポートする和音を，三和音，7th コード，9th コード，11th コードに限定した理由である．

	三和音 (5th)		7th まで		9th まで		11th まで		13th まで	
	T.M.	S.M.	T.M.	S.M.	T.M.	S.M.	T.M.	S.M.	T.M.	S.M.
ラテン	60.6	66.4	79.4	82.4	83.6	86.8	**84.1**	**87.4**	76.4	82.5
クラシック	71.7	72.4	86.5	86.0	**87.0**	**86.4**	86.8	86.3	85.0	85.6
ポップ	79.7	76.2	86.1	85.2	87.2	86.8	**87.4**	**87.2**	85.4	84.0
ジャズ	53.4	52.4	69.0	68.2	72.9	72.0	**73.1**	**72.3**	67.9	66.0
合計	63.3	63.7	77.6	77.4	80.4	80.6	**80.7**	**80.9**	76.2	76.2

表 A.2: 対応する和音型による精度の変化 (和音の根音により評価し，百分率で示した)．T.M. は分解能平均 (Tick Mean) を S.M. は楽曲平均 (Song Mean) を表す．トライアド，7th，9th，11th コードにおいて精度は最高となった．

A.4.3.2 根音推測の比較

本提案手法と *Melisma* の比較は表 A.3 のようになった．ジャズ楽曲について *Melisma* は (tick 平均スコアのみ) より高精度であったものの，ポップ，クラシック，ラテンにおいては提案システムがより良い結果を残した．同様に，ジャズにおいても楽曲平均の場合はそうである．その差異はクラシックとラテンにおいて顕著である．全体的な結果としては，提案手法は *Melisma* とほぼ同程度であり，差異は非常に小さいと言える (全体の分解能平均で 1%弱，全体の楽曲平均で 2%弱)．

これらの結果は別の方法でも分析可能である．図 A.9 は，算出された精度ごとの楽曲の数を示す柱状図となっている．二つの統計データは類似しているが，まったく同一ではない．*Melisma* が与えられた楽曲の精度としてつけた最低のスコアは 41.9%であるが，我々のシステムは 36.5%となっている．最高スコアは共に 100%に到達しているが，それぞれのシステムは別の曲に対してそれをマークした．155 の楽曲を対象として，我々のシステムは 102 の楽曲で 80%以上の精度をもったが，*Melisma* は 92 楽曲である．さらに注目に値するところとして，二つのシステムの求めた精度の差が最大となった三つの楽曲にお

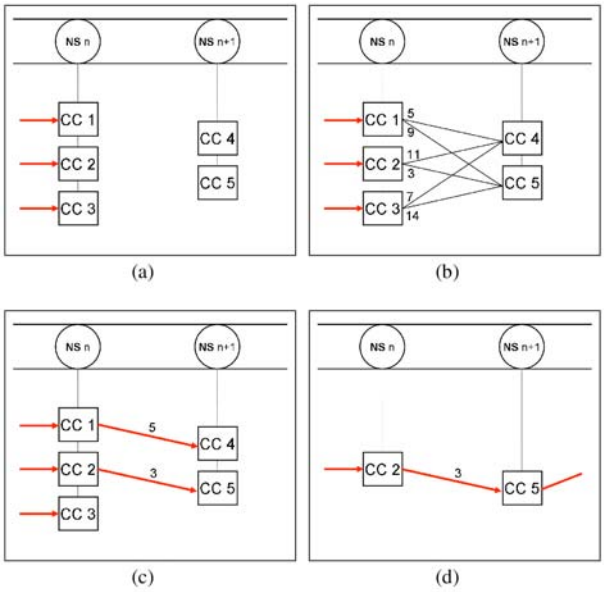

図 A.6: 処理全体の図示．二つの連続したノート・セグメント (NS) 間のグラフ構築: 適合する和音候補 (CC) の列挙 (**a**), 重み付きエッジによる接続 (**b**). 動的処理: 各和音候補に対する一つのエッジが保持される (**c**). 最後に，最適経路が選択される (**d**).

	本システム		Melisma	
	T.M.	S.M.	T.M.	S.M.
ラテン	**87.0**	**89.1**	85.2	86.1
クラシック	**86.1**	**85.5**	80.7	80.3
ポップ	**89.1**	**89.5**	88.5	88.8
ジャズ	76.9	**77.3**	**77.6**	75.7
合計	**83.4**	**84.1**	82.9	82.2

表 A.3: Band-In-A-Box データベースでの結果．精度は百分率である．T.M. は分解能平均 (Tick Mean) を S.M. は楽曲平均 (Song Mean) を表す．和音の根音のみを評価した．本システムは *Melisma* と同程度の精度を出した．

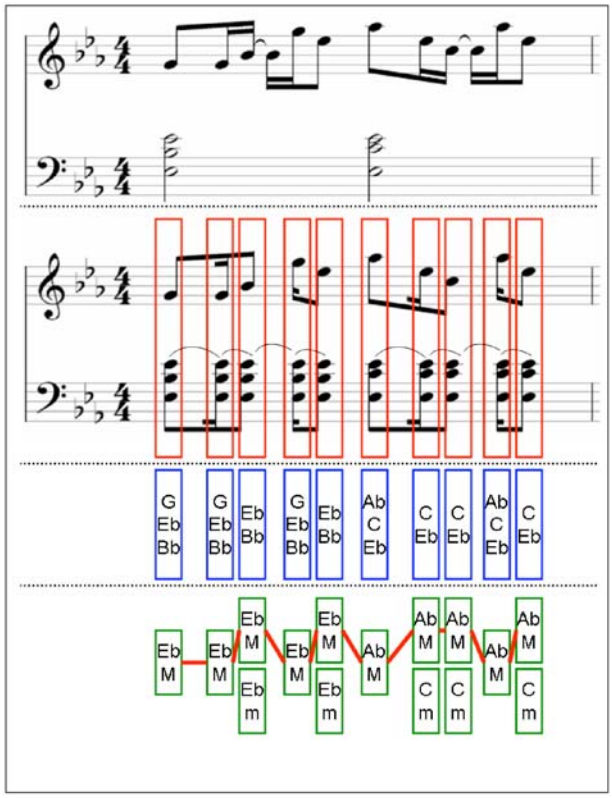

図 A.7: エルトン・ジョンの**僕の歌は君の歌** *(Your Song)* の第一小節を解析したもの．上部が原譜．下へ順に，楽譜へのホモリズミック変形の適用，ノート・セグメント，最後に和音候補．可読性のために和音候補は調のみ表示している．最適経路は互いにリンクしている和音により形成されている．

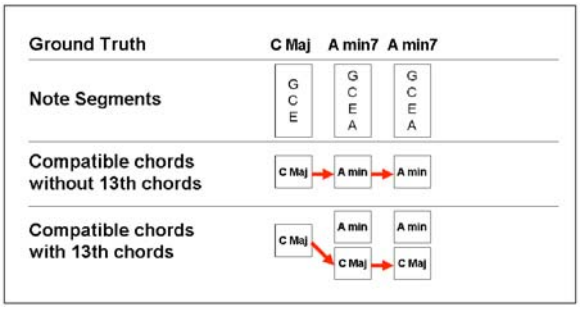

図 A.8: 対応する和音型を増やすことが誤認を増やす例．13th コードに対応していないときに，最終的に選択される経路は最適となる (正解データと一致する)．

図 A.9: 提案システム (上) と *Melisma*(下) の精度ごとのファイル数を示した柱状図．二つのシステムは異なる情報を導いていることがわかる．

いて，全て本システムが良い解析結果を残した: *April Joy* (90.0%と 51.2%), *A Taste of Honey* (86.4%と 42.1%), そして *April Showers* (95.3%と 75.6%) である．

これらの統計は，比較された二つの手法は同じくらいのスコアを出したものの，異なる種類の情報を用いるだろうということを示している．Melisma の拍解析と装飾音の使用は，二つのシステムにおける大きな差異であり，それにより二つの手法が結果が異なるかを説明できるだろう．

A.4.3.3　根音とモードの推定

Table 4 は，検知した和音の根音とモードを評価したときの本システムの結果である．ポップ，ラテン，クラシックのデータベースにおいて，結果は *Melisma* を根音のみで評価したときをわずかに上回った (83.3%)．楽曲平均で計算された全体的な結果についても *Melisma* より若干よい．すなわち，本システムは，精度が Melisma とほぼ同等であり，それぞれの検知した和音の根音だけでなくモードも提供できるといえる．

	本システム	
	分解能平均	楽曲平均
ラテン	86.8	88.9
クラシック	84.2	83.2
ポップ	88.8	89.1
ジャズ	75.3	76.0
合計	82.5	83.3

表 A.4: Band-In-A-Box データベースにおける結果．精度は百分率である．和音の根音とモードを評価した．本システムは根音のみを評価した *Melisma* と同程度の精度を見せた．

A.5　結論と今後の課題

我々は記号的多声データから和音推測を行う新しい手法を提案した．この手法は，非均一的時間分割，動的計画法，Lerdahl の和音遷移コストと，それぞれのノート・セグメントに対すルールベースの和音候補列挙を取り込んでいる．提案手法の精度を評価するため実験を行い，提案システムと *Melisma Music Analyzer* との比較を提示した．その結果から根音のみの場合，提案手法はわずかに精度が良く，検知された和音の根音とモードを評価した場合，*Melisma* の根音検知と同等の精度であった．

より多くの数の和音型をサポートする，ルール・セットの増加を考慮する，和音候補のグラフで最適経路を選択する別の方法を選択する，或いは，和音遷移コストとして別の距離を使用するなど，提案システムは拡張ができるだろうことから，これらの結果は見込み

のあるものだと考える．適合調は各ノート・セグメントごとに決定されるため，[4] で示されるように調の検知の仕方を変更することも計画されている．調変更についての注釈をともなった記号的楽譜の大きなデータベースが評価には必要であろう．各和音はシステムによって与えられた型とともに扱うこともできるので，和音の型の評価もまた可能である．音楽情報検索 (MIR) への可能な応用と音声和音認識への適応も進行中にある．

謝辞

この研究は，French National Research Agency (ANR) の支援による SIMBALS project の一部であり (JC07-188930)，また Aquitanine Regional Council にサポートされている．

参考文献

[1] R. Bellman, Dynamic Programming. Princeton University Press, 1957.

[2] J. Bello, “Audio-based Cover Song Retrieval using Approximate Chord Sequences: Testing Shift, Gaps, Swaps, and Beats,” in Proc. of the 8th International Conference on Music Information Retrieval (ISMIR), Vienna, Austria, 2007, pp. 239–244.

[3] A. M. Birch and O. Anon, Band-In-A-Box Independent Users Group File Archive 2, Available: `http://groups.yahoo.com/group/Band-in-a-Box-Files2/`, 2004.

[4] E. Chew, “The Spiral Array: An Algorithm for determining Key Boundaries,” in Proc. of the Second International Conference ICMAI 2002, Springer, 2002, pp. 18–31.

[5] G. Cabral and F. Pachet and J. Briot, “Automatic X Traditional Descriptor Extraction: The Case of Chord Recognition,” in Proc. of the 6th International Conference on Music Information Retrieval (ISMIR), London, U.K., 2005, pp. 444-449.

[6] P. Hanna, M. Robine, P. Ferraro, and J. Allali, “Improvements of Alignment Algorithms for Polyphonic Music Retrieval,” in Proc. of the CMMR08, International Symposium on Computer Music Medeling and Retrieval, Copenhagen, Denmark, 2008, pp. 244–251.

[7] P. Illescas, D. Rizo, and J. M. Iesta, “Harmonic, Melodic, and Functional Automatic Analysis,” in Proc. of the International Retrieval (ISMIR), Copenhagen, Denmark, 2007, pp. 165–168.

[8] J. P. Bello and J.Pickens, “A Robust Mid-Level Representation for Harmonic Content in Music Signals,” in Proc. of the 6th International Conference on Music Information Retrieval (ISMIR), London, U.K., 2005, pp. 304–311.

[9] K. Lee and M. Stanley, “Acoustic Chord Transcription and Key Extraction from Audio Using Key-Dependent HMMs Trained on Synthesized Audio,” The IEEE Transactions on Audio, Speech and Language Processing, vol. 16, no. 2, pp. 291–301, 2008.

[10] F. Lerdahl, Tonal Pitch Space. Oxford University Press, 2001.

[11] J.-F. Paiement, D. Eck, and S. Bengio, “A Probabilistic Model for Chord Progressions,” in Proc. of the 6th International Conference on Music Information Retrieval (ISMIR), London, U.K., 2005, pp. 312–319.

[12] H. Papadopoulos and G. Peeters, “Large-scale Study of Chord Estimation Algorithms Based on Chroma Representation and HMM,” in Proc. of the 5th International Conference on Content-Based Multimedia Indexing, Bordeaux, France, 2007, pp. 53–60.

[13] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in Readings in speech recognition, 1990, pp. 267–296.

[14] C. Rhodes, D. Lewis, and D. Müllensiefen, "Bayesian Model Selection for Harmonic Labelling," Mathematics and Computation in Music, 2007.

[15] A. Sheh and D. P. W. Ellis, "Chord Segmentation and Recognition Using EM-Trained Hidden Markov Models," in Proc. of the 4th International Conference on Music Information Retrieval (ISMIR), Baltimore, U.S.A., 2003, pp. 183–189.

[16] D. Temperley and D. Sleator, "The Melisma Music Analyzer," `http://www.link.cs.cmu.edu/music-analysis/`.

[17] D. Temperley, The Cognition of Basic Musical Structures. The MIT Press, 1999.