An Idiom-independent Representation of Chords for Computational Music Analysis and Generation

First Author
Affiliation1
author1@smcnetwork.org

Second Author
Affiliation2
author2@smcnetwork.org

Third Author
Affiliation3
author3@smcnetwork.org

ABSTRACT

In this paper we focus on issues of harmonic representation and computational analysis. A new idiom-independent representation is proposed of chord types that is appropriate for encoding tone simultaneities in any harmonic context (such as tonal, modal, jazz, octatonic, atonal). The *General Chord Type (GCT)* representation, allows the re-arrangement of the notes of a harmonic simultaneity such that abstract idiom-specific types of chords may be derived; this encoding is inspired by the standard roman numeral chord type labeling, but is more general and flexible. Given a consonance-dissonance classification of intervals (that reflects culturally-dependent notions of consonance/dissonance), and a scale, the GCT algorithm finds the maximal subset of notes of a given note simultaneity that contains only consonant intervals; this maximal subset forms the base upon which the chord type is built. The proposed representation is ideal for hierarchic harmonic systems such as the tonal system and its many variations, but adjusts to any other harmonic system such as post-tonal, atonal music, or traditional polyphonic systems. The GCT representation is applied to a small set of examples from diverse musical idioms, and its output is illustrated and analysed showing its potential, especially, for computational music analysis & music information retrieval.

1. INTRODUCTION

There exist different typologies for encoding note simultaneities that embody different levels of harmonic information/abstraction and cover different harmonic idioms. For instance, for tonal musics, chord notations such as the following are commonly used: figured bass (pitch classes denoted above a bass note – no concept of 'chord'), popular music guitar style notation or jazz notation (absolute chord), roman numeral encoding (relative to a key) [1]. For atonal and other non-tonal systems, pc-set theoretic encodings [2] may be employed.

A question arises: is it possible to devise a 'universal' chord representation that adapts to different harmonic idioms? Is it possible to determine a mechanism that, given some fundamental idiom features, such as pitch hierarchy and consonance/dissonance classification, can automatically encode pitch simultaneities in a pertinent manner for the idiom at hand?

Before attempting to answer the above question one could ask: What might such a 'universal' encoding system be useful for? Apart from music-theoretic interest and cognitive considerations/implications, a general chord encoding representation may allow developing generic harmonic systems that may be adaptable to diverse harmonic idioms, rather than designing ad hoc systems for individual harmonic spaces. This was the primary aim for devising the *General Chord Type (GCT)* representation. In the case of the project C-------T (name concealed for peer reviewing) [3], a creative melodic harmonisation system is required that relies on conceptual blending between diverse harmonic spaces in order to generate novel harmonic constructions; mapping between such different spaces is facilitated when the shared generic space is defined with clarity, its generic concepts are expressed in a general and idiom-independent manner, and a common general representation is available.

In recent years, many melodic harmonisation systems have been developed, some rule-based [4,5] or evolutionary approaches that utilize rule based fitness evaluation [6, 7] others relying on machine learning techniques like probabilistic approaches [8,9] and neural networks [10], grammars [11] or hybrid systems (e.g. [12]). Almost all of these systems model aspects of tonal harmony: from "standard" Bach–like chorale harmonisation [4,10] among many others) to tonal systems such as "classic" jazz or pop ([9,11] among others). These systems aim to produce harmonizations of melodies that reflect the style of the discussed idiom, which is pursued by utilising chords and chord annotations that are characteristic of the idiom. For instance, the chord representation for studies in the Bach chorales include usually standard Roman numeral symbols, while jazz approaches encompass additional information about extensions in the guitar style encoding.

For tonal computational models, Harte's representation [13] provides a systematic, context-independent syntax for representing chord symbols which can easily be written and understood by musicians , and, at the same time, is simple and unambiguous to parse with computer programs. This chord representation is very useful for annotating manually tonal music - mostly genres such as pop, rock, jazz that use guitar-style notation. However, it

cannot be automatically extracted from chord reductions and is not designed to be used in non-tonal musics.

In this paper, firstly, we present the main concepts behind the *General Chord Type* representation and give an overall description, then, we describe the GCT algorithm that automatically computes chord types for each chord, then, we present examples form diverse music idioms that show the potential of the representation and give some examples of applying statistical learning on such a representation, and, finally, we will discuss problems and future improvements.

2. REPRESENTING CHORDS

Harmonic analysis focuses on describing the harmonic content of pitch collections/patterns within a given music context in terms of harmonic labels, classes, functions and so on. Harmonic analysis is a rather complex musical task that involves not only finding roots and labelling chords within a key, but also segmentation (points of harmonic change), identification of non-chord notes, metric information and more generally musical context [14]. In this paper, we focus on the core problem of labelling chords within a given pitch hierarchy (e.g. key); thus we assume that a full harmonic reduction is available as input to the model (manually constructed harmonic reductions).

Our intention is to create an analytic system that may label *any* pitch collection, based on a set of user-defined criteria rather than on standard tonal music theoretic models or fixed psychoacoustic properties of harmonic tones. We intend our representation to be able to cope with chords not only in the tonal system, but any harmonic system (e.g. octatonic, whole-tone, atonal, traditional harmonic systems, etc.).

Root-finding is a core harmonic problem addressed primarily following two approaches: the standard stack-of-thirds approach and the virtual pitch approach. The first attempts to re-order chord notes such that they are separated by (major or minor) third intervals preserving the most compact ordering of the chord; these stacks of thirds can then be used to identify the possible root of a chord (see, for instance, recent advanced proposal by [15]). The second approach, is based on Terhard's virtual pitch theory [16] and Parncutt's psychoacoustic model of harmony [17]; it maintains that the root of a chord is the pitch most strongly implied by the combined harmonics of all its constituent notes (intervals derived from the first members of the harmonic series are considered as 'root supporting intervals').

Both of these approaches rely on a fixed theory of consonance and a fixed set of intervals that are considered as building blocks of chords. In the culture-sensitive stack-of-thirds approach, the smallest consonant intervals in tonal music, i.e. the major and minor thirds, are the basis of the system. In the second 'universal' psychoacoustic approach, the following intervals, in decreasing order of importance, are employed: unison, perfect fifth, major third, minor seventh, and major second. Both of these approaches are geared towards tonal harmony, each with its strengths and weaknesses (for instance, the second approach has an inherent difficulty with minor harmonies). Neither of them can be readily extended to other idiosyncratic harmonic systems.

Harmonic consonance/dissonance has two major components: Sensory-based dissonance (psychoacoustic component) and music-idiom-based dissonance (cultural component)[18]. Due to the music-idiom dependency component, it is not possible to have a fixed universal model of harmonic consonance/dissonance. A classification of intervals into categories across the dissonance-consonance continuum can be made only for a specific idiom. The most elementary classification is into two basic categories: consonant and dissonant. For instance, in the common-practice tonal system, unisons, octaves, perfect fifths/fourths (perfect consonances) and thirds and sixths (imperfect consonances) are considered to be consonances, whereas the rest of the intervals (seconds, sevenths, tritone) are considered to be dissonances; in polyphonic singing from Epirus, major seconds and minor sevenths may additionally be considered 'consonant' as they appear in metrically strong positions and require no resolution; in atonal music, all intervals may be considered equally 'consonant'.

Let's examine the case of tonal and atonal harmony; these are probably as different as two harmonic spaces may be. In the case of tonal and atonal harmony, some basic concepts are shared; however, actual systematic descriptions of chord-types and categories are drastically different (if not incompatible), rendering any attempt to 'align' two input spaces challenging and possibly misleading (Figure 1). On one hand, tonal harmony uses a limited set of basic chord types (major, minor, diminished, augmented) with extensions (7ths, 9ths etc.) that have roots positioned in relation to scale degrees and the tonic, reflecting the hierarchic nature of tonal harmony; on the other hand, atonal harmony employs a flat mathematical formalism that encodes pitches as pitch-class sets leaving aside any notion of pitch hierarchy, tone centres or more abstract chord categories and functions. It seems as if it is two worlds apart having as the only meeting point the fact that tones sound together (physically sounding together or sounding close to one another allowing implied harmony to emerge).

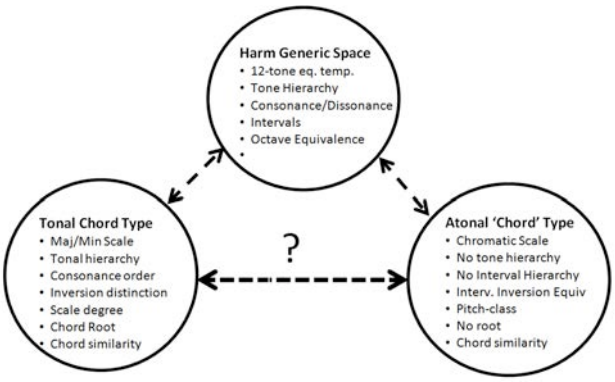

Figure 1. Is mapping between 'opposing' harmonic spaces possible?

Pc-set theory of course, being a general mathematical formalism, can be applied to tonal music, but, then its descriptive potential is mutilated and most interesting tonal harmonic relations and functions are lost. For in-

stance, the distinction between major and minor chords is lost if Forte's *prime form* is used (037 for both - these two chord have identical interval content), or a dominant seventh chord is confused with half-diminished seventh (prime form 0258); even, if *normal order* is used, that is less general, for the dominant seventh (0368), the root of the chord is not the 0 on the left of this ordering (pc 8 is the root). Pitch-class set theory is not adequate for tonal music. At the same time, the roman-numeral formalism is inadequate for atonal music as major/minor chords and tonal hierarchies are hardly relevant for atonal music.

In trying to tackle issues of tonal hierarchy, we have devised a novel chord type representation, namely the *General Chord Type (GCT)* representation, that takes as its starting point the common-practice tonal chord representation (for a tonal context, it is equivalent to the standard roman-numeral harmonic encoding), but is more general as it can be applied to other non-standard tonal systems such as modal harmony and, even, atonal harmony. This representation draws on knowledge from the domain of psychoacoustics and music cognition, and, at the same time, 'adjusts' to any context of scales, tonal hierarchies and categories of consonance/dissonance.

At the heart of the GCT representation is the idea that the 'base' of a note simultaneity should be consonant. The GCT algorithm tries to find a maximal subset that is consonant; the rest of the notes that create dissonant intervals to one or notes of the chord 'base' form the chord 'extension'. The GCT representation has common characteristics with the stack-of-thirds and the virtual pitch root finding methods for tonal music, but has differences as well (see section 4.3). Moreover, the user can define which intervals are considered 'consonant' giving thus rise to different encodings. As will be shown in the next sections, the GCT representation encapsulates naturally the structure of tonal chords and at the same time is very flexible and can readily be adapted to different harmonic systems.

3. THE *GENERAL CHORD TYPE* REPRESENTATION

3.1 Description of the GCT Algorithm

Given a classification of intervals into consonant/dissonant (binary values) and an appropriate scale background (i.e. scale with tonic), the *GCT algorithm* computes, for a given multi-tone simultaneity, the 'optimal' ordering of pitches such that a maximal subset of consonant intervals appears at the 'base' of the ordering (left-hand side) in the most compact form. Since a tonal centre (key) is given, the position within the given scale is automatically calculated.

Input to the algorithm is the following:

- Consonance vector: The user defines which intervals are consonant/dissonant. A 12-point vector is employed where each vector entry corresponds to a pitch interval from 0 to 11 - in the current version of the algorithm, Boolean values are used (i.e., consonant=1, dissonant=0). For instance, the vector [1,0,0,1,1,1,0,1,1,1,0,0] means that the unison, minor and major third, perfect fourth and fifth, minor and major sixth intervals are consonant – dissonant intervals are the seconds, sevenths and the tritone; this specific vector is referred to in this text as the common-practice consonance vector.
- Pitch Scale Hierarchy: The pitch hierarchy (if any) is given in the form of scale tones and a tonic. For instance, a D maj scale is given as: 2, [0,2,4,5,7,9,11], or an A minor pentatonic scale as: 9, [0,3,5,7,10].
- Input chord: list of MIDI pitch numbers (converted to pc-set).

GCT Algorithm (core) - computational pseudocode

Input: (i) the pitch scale (tonality), (ii) a vector of the intervals considered consonant, (iii) the pitch class set (pc-set) of a note simultaneity
Output: The roots and types of the possible chords describing the simultaneity

1. find all maximal subsets of pairwise consonant tones
2. select maximal subsets of maximum length
3. **for** all selected maximal subsets **do**
4. order the pitch classes of each maximal subset in the most compact form (chord 'base')
5. add the remaining pitch classes (chord 'extensions') above the highest of the chosen maximal subset's (if necessary, add octave - pitches may exceed the octave range)
6. the lowest tone of the chord is the 'root'
7. transpose the tones of the chord so that the lowest becomes 0
8. find position of the 'root' in regards to the given tonal centre (pitch scale)
9. **endfor**

The GCT algorithm encodes most chord types 'correctly' in the standard tonal system. In example 1, Table 1 the note simultaneity [C,D,F#,A] or [0,2,6,9] in a G major key is interpreted as [7,[0,4,7,10]], i.e. as a dominant seventh chord (see similar example in Section 3.3).

However, the algorithm is undecided in some cases, and even makes 'mistakes' in other cases. In most instances of multiple encodings, it is suggested that these ideally should be resolved by taking into account other harmonic factors (e.g., bass line, harmonic functions, tonal context, etc.). For instance, the algorithm gives two possible encodings for a [0,2,5,9] pc-set, namely minor seventh chord or major chord with sixth (see Table1, example 2); such ambiguity may be resolved if tonal context is taken into account. For the [0,3,4,7] pc-set with root 0, the algorithm produces two answers, namely, a major chord with extension [0,[0,4,7,15]] and a minor chord with extension [0,[0,3,7,16]]; this ambiguity may be resolved if key context is taken into account: for instance, [0,4,7,15] would be selected in a C major or G major context and [0,3,7,16] in a C minor or F minor context. Symmetric chords, such as the augmented chord or the diminished seventh chord, are inherently ambiguous; the algorithm suggests multiple encodings which can be resolved only by taking into account the broader harmonic context (see Table1, example 3).

		Example 1	*Example 2*	*Example 3*
	Tonality - key	G: [7, [0, 2, 4, 5, 7, 9, 11]]	C: [0, [0, 2, 4, 5, 7, 9, 11]]	C: [0, [0, 2, 4, 5, 7, 9, 11]]
	Cons. Vector	[1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0]	[1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0]	[1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0]
	Input pc-set	[60, 62, 66, 69, 74] [0, 2, 6, 9]	[50, 60, 62, 65, 69] [0, 2, 5, 9]	[62, 68, 77, 71] [2, 5, 8, 11]
	Maximal subsets	[2, 6, 9]	[2, 5, 9] and [5, 9, 0]	[2, 5], [5, 8], [8, 11], [2, 11]
	Narrowest range	[2, 6, 9]	[2, 5, 9] and [5, 9, 0]	[2, 5], [5, 8], [8, 11], [2, 11]
	Add extensions	[2, 6, 9, **12**]	[2, 5, 9, **12**] and [5, 9, 0, **14**]	all rotations of [2,5,8,11]
	Lowest is root	2 (note D)	2 and 5 (notes D & F)	2,5,8,11 (resp. for each rotation)
	Chord in root position	[2, [0, 4, 7, 10]]	[2, [0, 3, 7, 10]] & [5, [0, 4, 7, 9]]	[X,[0,3,6,9]], where X∈{2,5,8,11}
	Relative to key	**[7, [0, 4, 7, 10]]**	**[2, [0, 3, 7, 10]] & [5, [0, 4, 7, 9]]**	**[X,[0,3,6,9]], where X∈{2,5,8,11}**
Extra steps:	Subset overap		**[2, [0, 3, 7, 10]]**	
	Base in scale			**[11,[0,3,6,9]]**

Table 1. Examples of applying the GCT algorithm.

Since the aim of this algorithm is not to perform sophisticated harmonic analysis, but rather to find a practical and efficient encoding for tone simultaneities (to be used, for instance, in statistical learning and automatic harmonic generation – see end of Section 4), we decided to extend the algorithm so as to reach in every case a single chord type for each simultaneity (no ambiguity).

***GCT Algorithm* (additional steps)** - for unique encoding

If more than one maximal subsets exist:

- Overlapping of maximal subsets: create a sequence of maximal subsets by ordering them so as to have maximal overlapping between them and keep the maximal subset that appears first in the sequence (chord's base)
- Chord base notes are scale notes: prefer maximal subset that contains only pcs that appear in the given scale (tonal context) – i.e. avoid non-scale notes in the chord base (this rule is rather arbitrary and is under consideration)
- if neither of the above give a unique solution, chose one encoding at random

Additional adjustment: for dyads, in a tonal context, prefer perfect fifth over perfect fourth, and prefer seventh to second intervals

The additional steps select chord type [2, [0,3,7,10]] in example 2, Table1 (maximal overlapping between two maximal subsets), and [11, [0,3,6,9]] in example 3, Table 1 (last pitch-class is Ab that is a non-scale degree in C major).

3.2 Formal description of the Core GCT Algorithm

The proposed algorithm for extracting the computation of GCT receives a simultaneity of pitches that are transformed into pitch classes and produces a chord type relative to a key, namely the *root*, the *base* and the *extension*, which specify qualitative information about the chord that more precisely describes this simultaneity. A detailed description of the algorithm follows, based on an example input simultaneity. Suppose that the input set of notes results in the pc-set [0, 2, 6, 9], which could be described as a D major chord with minor seventh regarding the tonal music environment – described by the υ = [1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0] consonance vector. Therefore, the algorithm should produce an output in the form: [r, [b], [e]] = [2, [0, 4, 7], [10]].

By utilising the input pc-set and given a consonance vector that represents a selected music idiom (in this example the consonance vector is υ = [1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0]), a binary matrix is constructed that is denoted as *B*. Each row and column of *B* represents a pitch class of the input chord, while a matrix entry is 1 or 0, signifying whether the pair of row and column pcs are consonant or dissonant respectively – according to the current consonance vector. Strictly, if the consonance vector is denoted as υ and the input pcset as *p*, then ∀ i, j ∈ {1, 2, . . ., length(*p*)}

$$B_{ij} = c_{|p_i - p_j|} \tag{1}$$

where the function length(*x*) return the length of vector *x*. The *B* matrix in the discussed example, where *p* = [0, 2, 6, 9], is the following:

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \tag{2}$$

Afterwards, a tree is constructed for each of the rows of *B*. The root node of these trees is the pitch class that corresponds to the respective row, while their branches from leaves to nodes include pitch classes that are pairwise consonant (according to υ). The construction of the tree that corresponds to the *i*–th element of *p*, is implemented by recursively traversing *B* in a depth–first–search (DFS) fashion, beginning from the *i*–th row and following the paths 'circumscribed' by the occurrences of units. Such a traversal is exhibited in Table 2 for the second row of the current example's *B* matrix. This step's outcome is a collection of trees, each of which corresponds to a row of *B*. The trees of the current example are shown in Table 3.

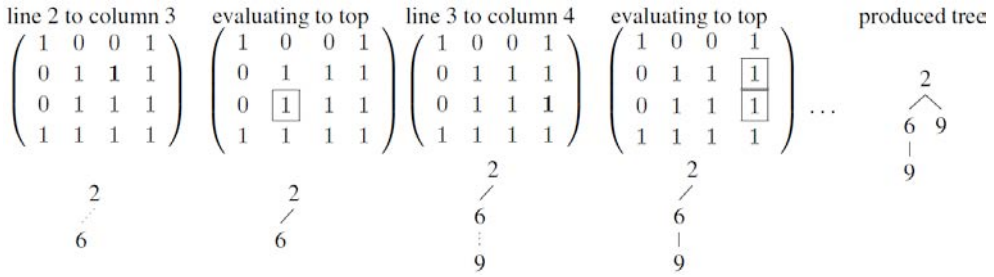

Table 2. The steps of the algorithm when scanning the path of the second row.

line 1	line 2	line 3	line 4
0	**2**	6	
|	/ \	|	9
9	**6** 9	9	
	|		
	9		

Table 3. All the trees for the current example. The maximal path is highlighted with boldface typesetting.

After the application of the above procedure, the paths from root to leaves with maximal length are kept either as the output chord candidates, or for further processing in the steps described in the remaining of this section. In the current example there is a single maximal path ([2, 6, 9]), which is highlighted with boldface typesetting (Table 3). After the longest path has been extracted, the pitch classes that constitute it, are recombined in their most compact form, which in the current example is [2, 6, 9] (unaltered). The pitch class 0 of the initial [2, 6, 9] pc-set is considered as an *extension*. Thereby, the simultaneity [0, 2, 6, 9] is circularly shifted to [2, 6, 9, 12], disregarding the fact that pitch classes can take integer values between 0 and 11. In turn, [2, 6, 9, 12] is transformed to the following [r, [b], [e]] denotation: [2, [0, 4, 7], [10]]. This denotation clarifies that the simultaneity [0, 2, 6, 9] is actually a major chord (base [0, 4, 7]) with a minor seventh (extension [10]) and fundamental pitch class 2, (i.e. D7). As the tonal context is given as input, for instance G major key, the absolute chord type [2, [0,4,7,10]] (i.e. D7 chord) is converted to relative chord type, i.e., [7,[0,4,7,10]] which means dominant seventh in G major. This is equivalent to the roman numeral analytic types.

3.3 An example analysis with GCT

An example harmonic analysis of a Bach Chorale phrase illustrates the proposed GCT chord representation (Figure 2). For a tonal context, chord types are optimised such that pcs at the left hand side of chords contain only consonant intervals (i.e. 3^{rds} & 6^{ths}, and Perfect 4^{ths} & 5^{ths}). For instance, the major 7th chord is written as [0,4,7,10] since set [0,4,7] contains only consonant intervals whereas 10 that introduces dissonances is placed on the right-hand side – this way the relationship between major chords and major seventh chords remains rather transparent and is easily detectable. Within the given D major key context it is simple to determine the position of a chord type in respect to the tonic – e.g. [7,[0,4,7,10]] means a major seventh chord whose root is 7 semitones above the tonic, amounting to a dominant seventh. This way we have an encoding that is analogous to the standard roman numeral encoding (Figure 2, top row). If the tonal context is changed, and we have a chromatic scale context (arbitrary 'tonic' is 0, i.e. note C) and we consider all intervals equally 'consonant', we get the second GCT analysis in Figure 1 which amounts to normal orders (not prime forms) in a standard pc-set analysis – for tonal music this pc-set analysis is weak as it misses out important tonal hierarchical relationships (notice that the relation of the dominant seventh chord type to the plain dominant chord is obscured). Note that relative 'roots' to the 'tonic' 0 are preserved as they can be used in harmonic generation tasks.

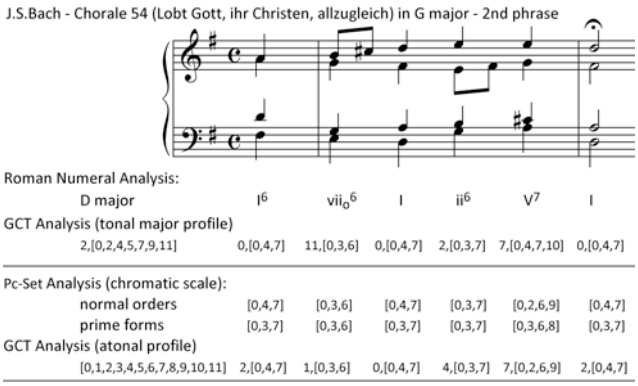

Figure 2 Chord analysis of a Bach Chorale phrase by means of traditional roman numeral analysis, pc-sets and two versions of the GCT algorithm.

For practical reasons of space in the musical illustrations, the form [r,[b],[e]] is not preserved: the base and extension is concatenated and brackets are omitted. For instance: [7,[0,4,7],[10]] may be depicted as 7,[0,4,7,10] or even as 7.04710.

4. HARMONIC ENCODING & ANALYSIS WITH THE GCT

The GCT algorithm has been applied to tonal extracts from standard tonal pieces, such as Bach Chorales, but additionally it has been tested out on harmonic structures from diverse harmonic idioms. Some examples are presented below to give an idea of the potential of the GCT representation. Strong points of the encoding are given along with weaknesses. Some aspects of the analysis are difficult to judge in some idioms and further study in required.

4.1 GCT Encoding Examples

In common-practice tonal music, GCT works very well. Mistakes are sometimes made in case of symmetric chords such as the diminished seventh chord or the augmented triad. In the case of the half diminished seventh chord GCT 'prefers' to label it as a minor chord with added sixth instead of a diminished chord with minor seventh. Chords that include chromatic notes such as the German sixth, Italian sixth, Neapolitan sixth are encoded consistently even though not necessarily coinciding with analytic interpretations by theorists (the French sixth is more tricky as it is a symmetric chord and GCT finds two equally prominent 'roots').

Below, a number of examples are presented that illustrate the application of the GCT algorithm on diverse harmonic textures. The first example (Figure 3) is taken from the first measures of Beethoven's *Moonlight Sonata*. In this example, GCT encodes classical harmony in a straightforward manner. All instances of the tonic chord inverted or not (i.e., C# minor) are tagged as 0,[0,3,7] and [10] is added when the 7th is present; the dominant seventh is 7,[0,4,7,10] and it appears once without the fifth [7]; the fourth chord is a Neapolitan sixth and it is encoded as 1,[0,4,7] which means major chord on lowered second degree (Db major chord in the C# minor key).

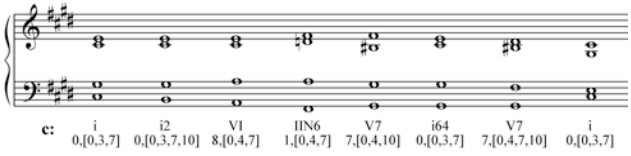

Figure 3 Beethoven, Sonata 14, op.27-2 (reduction of first five measures). Top row: roman numeral harmonic analysis; bottom row: GCT analysis. GCT successfully encodes all chords, including the Neapolitan sixth chord (fourth chord).

In the example of Figure 4 a tonal chord progression by G. Gershwin is presented. Chromaticism is apparent in this passage. The GCT 'agrees' with the roman numeral analysis of the excerpt including the Italian sixth chord that is labelled as 8,[0,4,10], and it even labels the chord that was left without a roman numeral tag by the analyst (see question mark) encoding it as a minor chord with sixth on the flattened sixth degree (Gb-Bbb-Db-Eb) (Note: actually it could be even encoded as a half-diminished 7th on the fourth degree Eb-Gb-Bbb-Db).

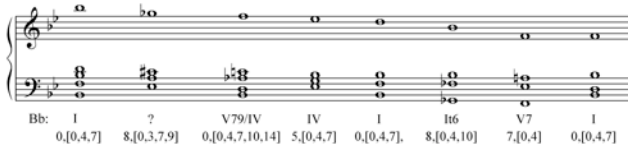

Figure 4. G. Gershwin, *Rhapsody in Blue* (reduction of first five measures). Top row: roman numeral harmonic analysis; bottom row: GCT analysis. GCT successfully identifies all chords (see text).

Figure 5 illustrates an Early Renaissance example of fauxbourdon by G. Dufay. Parallel motion of voices is typical in this idiom. The GCT labels correctly all dyads and triads, taking into account musica ficta that produces rather unusual chord progressions in regards to standard tonal harmony.

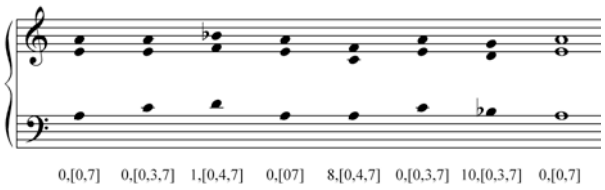

Figure 5. G. Dufay's *Kyrie* (reduction) - first phrase in A phrygian mode that exemplifies parallel motion in fauxbourdon and a phrygian cadence (early Renaissance). GCT correctly identifies and labels the open fifths as well as the triadic chords.

In Figure 6 an example from the polyphonic singing tradition of Epirus is presented. This very old 2-voice to 4-voice polyphonic singing tradition is based on the anhemitonic pentatonic pitch collection and more specifically the pentatonic minor scale that functions as source for both the melodic and harmonic content of the music. A unique harmonic aspect of these songs is the unresolved dissonances (major second and minor seventh intervals) at structurally stable positions of the pieces (e.g. cadences). In the example two GCT versions are presented: the first (top row) depicts the encoding for the standard consonance vector and the second (bottom row) presents the GCT labelling that considers additionally major seconds and minor sevenths as 'consonant' (it is the same as for the 'atonal' consonance vector as no minor seconds and major sevenths exist in the idiom). It is interesting to note that for the standard consonance vector almost all chords have the drone tone as their root. On the other hand, in the second encoding different relations between chords become apparent (e.g. 10,[0,2,5] and 10,[0,2,5,7]) and also an oscillation of the chord 'root' between the tonic and a note a tone lower is highlighted. Polyphonic songs from Epirus are the focus of a different study [19].

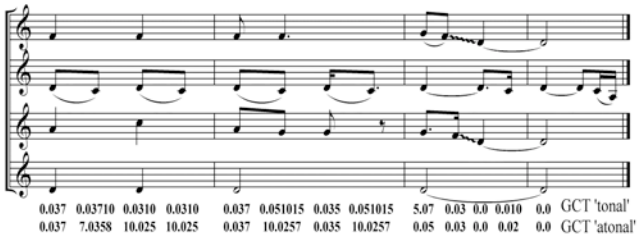

Figure 6 Excerpt from a traditional polyphonic song from Epirus. Top row: GCT encoding for standard common-practice consonance vector; bottom row: GCT encoding for atonal harmony – all intervals 'consonant' (this amounts to pc-set 'normal orders')

4.2 Learning and generation with GCT

In a current study, the GCT representation has been utilised in automatically analysing and encoding scores (actually, harmonic reductions of scores) from diverse idioms, and then employing this extracted information for melodic harmonisation. In [20] the authors discuss the utilization of a well–studied probabilistic methodology, namely, the hidden Markov model (HMM) methodology, in combination with constraints that incorporate fixed beginning and ending chords and intermediate anchor chords; to this end, a constrained HMM (CHMM) is developed. This work is motivated by the fact that the be-

ginning and the cadence of a phrase/piece is characteristic of its structural identity; such characteristic structural points in pieces can be modelled using higher-level hierarchical models (e.g. probabilistic grammars). Additionally, the CHMM methodology allows for the manual insertion of intermediate chords, providing alternative harmonisations that comply with specific constraints.

The reported results indicate that the CHMM method, harnessed with the novel General Chord Type (GCT) algorithm, functions effectively towards convincing melodic harmonisations in diverse idioms. In Figures 7 & 8, two examples of melodic harmonisation are illustrated for a Bach chorale melody and for a traditional melody from Epirus. In both cases, the system has been trained on a corpus of harmonic reductions of pieces in the idiom, and, then, used to generate new melodic harmonisations. The results are very good: the Bach chorale harmonisation is typical of the style and at the same time not trivial (uses secondary dominants that enrich the harmonisation); the Epirus melody harmonisation is close to the style of polyphonic singing (if additional melodic and rhythmic elements were added the phrase would become rather typical of the idiom).

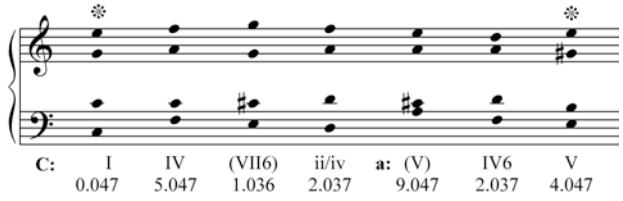

Figure 7. Automatically generated GCTs for a Bach Chorale melody employing a HMM for fixed boundaries (first and last chords are given). Voice leading has been arranged manually.

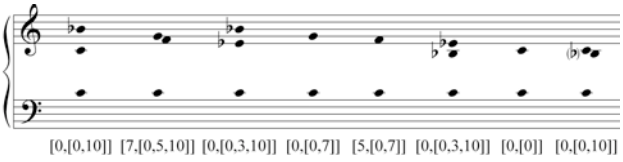

Figure 8. Automatically generated GCTs for an Epirus melody (reduced version) employing a HMM for fixed boundaries. Voice leading has been arranged manually.

4.3 Discussion and future development

The current version of GCT encodes only the chord type and the relative position of its 'root' to the local tonic of a given scale. However, it can readily be extended to incorporate explicit information on chord inversions (i.e. bass note position), on scale degrees (chromatic notes that do not belong to the current scale can be tagged so that indirectly scale degrees are indicated), and, even, on voice-leading (for instance, motion of bass, or even for note extensions that may require resolution by downwards step-wise motion). A rich chord representation should embody such information.

The organisation of tones by GCT for the 'standard' consonance vector gives results quite close to those produced by the stack-of-thirds technique, as implicit in the latter is consonance of thirds and fifths (as two thirds sum up to a fifth). Some difference are:

- the stack-of-thirds approach usually requires traditional note names (that allow enharmonic spellings) whereas the GCT is based on pitch classes (no direct explicit link to a scale). For instance, GCT considers the chord CEG# or CEAb ([0,4,8]) as consonant since its intervals are pairwise consonant[1], i.e. two 4 semitone intervals (major thirds) and one 8 semitone interval (minor sixth or augmented fifth) with root any one of the three tones; stack-of-thirds determines C as the root in the first case and Ab in the second case. The GCT algorithm misses out on sophisticated tonal scale information but is still informative at the same time being simpler, and easier to implement.
- in the standard consonance vector version of GCT, diminished fifths are not allowed whereas in the stack-of-thirds approach all fifths are allowed. For instance, the root of the half-diminished chord BDFA is B according to the stack-of-thirds whereas GCT considers D as the root and B as a sixth above the root (DFAB), i.e. diminished triads are not consonant chords according to CGT. Of course, the consonance vector in GCT may be altered so that the tritone is also consonant in which case the two approaches are closer.
- the stack-of-thirds method allows empty third positions in the lower part of the stack whereas GCT always prefers to have a compact consonant set of pitches at the bottom. For instance, a chord comprising of notes: CEFG ([0,4,5,7]) will be arranged as FCEG by the stack-of-thirds technique and CEGF ([0,4,7,17]) by GCT.

In relation to the virtual pitch root finding method, the proposed approach differs in that minor thirds are equally consonant to major thirds allowing equal treatment of major and minor chord (as opposed to the virtual pitch approach that is biased towards major thirds due to the structure of the harmonic series).

It is also possible to redesign the GCT algorithm altogether so as to make use of non-binary consonance/dissonance values allowing thus a more refined consonance vector. Instead of filling in the consonance vector with 0s and 1s, it can be filled with fractional values that reflect degrees of consonance derived from perceptual experiments (e.g., [21]) or values that reflect culturally-specific preferences. Such may improve the algorithm's performance and resolve ambiguities in certain cases (future work).

5. CONCLUSIONS

In this paper a new representation of chord types has been presented that adapts to diverse harmonic idioms allowing the analysis and labelling of tone simultaneities in any harmonic context. The *General Chord Type (GCT)*

[1] Question: why is the augmented triad considered dissonant when all its tones are pairwise consonant?

representation, allows the re-arrangement of the notes of a harmonic simultaneity such that idiom-specific types of chords may be derived. Given a consonance/dissonance classification of intervals (that reflects culturally-dependent notions of consonance/dissonance), and a (set of) scales, the GCT algorithm finds the maximal subset of notes of a given note simultaneity that contains only consonant intervals; this maximal subset forms the basis upon which the chord type is built. The proposed representation is ideal for hierarchic harmonic systems such as the tonal system and its many variations, but adjusts to any other harmonic system such as post-tonal, atonal music, or traditional polyphonic systems.

The GCT representation was applied to a small set of examples from diverse musical idioms, and its output was presented and analysed showing its potential use, especially, for computational music analysis and music information retrieval tasks. The encoding provided by GCT is not always correct according to the interpretation given by music theorists, but, at least, it is consistent (i.e. a certain chord will always be encoded the same way) rendering it adequate for machine learning and generation (e.g. melodic harmonisation) where music theoretical correctness is not so important. Sometimes GCT 'uncovers' chordal relations that are obscured by notation and enharmonic spellings, and may assist a musician in harmonic analysis. Overall, the proposed encoding seems to be promising and potentially useful in computational music applications.

Acknowledgments

The project C-------T (name concealed for peer reviewing) acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: XXXXXX. Special thanks are due to XXXXX XXXXX for preparing the harmonic dataset that has been used in the music examples in the paper.

6. REFERENCES

[1] Laitz, S. G. (2008). The complete musician: An integrated approach to tonal theory, analysis, and listening (Vol. 1). Oxford University Press, USA.

[2] Forte, A. (1973). *The structure of atonal music.* Yale University Press.

[3] Anonymous. (2014) for peer review

[4] Ebcioglu, K. (1988) An expert system for harmonizing four part chorales, *Computer Music Journal*, vol. 12, no. 3, pp. 43–51.

[5] Pachet F. and Roy, P. (2001) Musical harmonization with constraints: A survey. *Constraints*, vol. 6, no. 1, pp. 7–19, Jan. 2001.

[6] Phon-amnuaisuk S. and Wiggins, G.A. (1999) The four-part harmonisation problem: A comparison between genetic algorithms and a rule–based system," in *In proceedings of the AISB99 symposium on musical creativity*. AISB, 1999, pp. 28–34.

[7] Donnelly P. and Sheppard J. (2011) Evolving four-part harmony using genetic algorithms, in *Proceedings of the 2011 International Conference on Applications of Evolutionary Computation - Volume Part II*, ser. EvoApplications'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 273–282.

[8] Paiement, J.-F., Eck, D. and Bengio, S. (2006) Probabilistic melodic harmonization, in *Proceedings of the 19th International Conference on Advances in Artificial Intelligence: Canadian Society for Computational Studies of Intelligence*, ser. AI'06. Berlin, Heidelberg: Springer-Verlag, pp. 218–229.

[9] Simon, I. Morris, D. and Basu, S. (2008) Mysong: Automatic accompaniment generation for vocal melodies, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '08. New York, NY, USA: ACM, pp. 725–734.

[10] Hild, H. Feulner, J. and Menzel, W. (1991) HARMONET: A neural net for harmonizing chorales in the style of J. S. Bach. J. E. Moody, S. J. Hanson, and R. Lippmann, Eds. Morgan Kaufmann, 1991, pp. 267–274.

[11] Granroth-Wilding, M.T. (2013) Harmonic analysis of music using combinatory categorial grammar, Ph.D. dissertation, Institute for Language, Cognition and Computation School of Informatics University of Edinburgh, Edinburgh, Scotland, Nov. 2013.

[12] Chuan C.-H. and Chew E. (2007) A hybrid system for automatic generation of style-specific accompaniment, in *Proceedings of the 4th International Joint Workshop on Computational Creativity*. Goldsmiths, University of London.

[13] Harte, C., Sandler, M. B., Abdallah, S. A., & Gómez, E. (2005). Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations. In ISMIR (pp. 66-71).

[14] Temperley, D. (2001). *The cognition of basic musical structures.* MIT press.

[15] Sapp, C. S. (2007). Computational Chord-Root Identification in Symbolic Musical Data: Rationale, Methods, and Applications. *Computing in Musicology*, 15, pp. 99-119.

[16] Terhardt, E. (1974). Pitch, consonance and harmony. *Journal of the Acoustical Society of America*, 55, pp. 1061-1069.

[17] Parncutt, R. (1989). *Harmony: A psychoacoustical approach*. Springer-Verlag Publishing.

[18] Oxenham, A.J. (2013) The Perception of Musical Tones. In *The Psychology of Music*. Deutsch, D. (Ed.). Academic Press.

[19] Anonymous (2014a) for peer review

[20] Anonymous (2014b) for peer review

[21] William Hutchinson & Leon Knopoff (1978) The acoustic component of western consonance, *Interface*, 7:1, pp. 1-29.