The ontocomc Ontology: A Semantic Framework for Hindemith's Harmonic System

Thanasis Apostolopoulos[1], Spyridon Kantarelis[1] and Giorgos Stamou[1]

[1]*Artificial Intelligence and Learning Systems Laboratory, National Technical University of Athens, Heroon Polytechniou 9, Zografou, Attica, 157 80, Greece*

Abstract

Ontological representation of music concepts is a growing area of research in both the data science and computational musicology communities. This paper introduces the ontocomc ontology, a scheme that focuses on note simultaneities, sequences of chords and their automated semantic annotation. Our ontology is based on Paul Hindemith's harmonic theory in his Craft of Musical Composition, an original construction completely at odds with traditional tonal harmony. A key characteristic of this scheme is its handling all conceivable simultaneities in a uniform way. This harmonic system has received very little (if any) attention in past data science research; in this sense, our approach constitutes a major novelty, leveraging the representational power of ontologies applied to Hindemith's exhaustive classification of note simultaneities. Chord sequences and their constituent parts are enriched with semantic features enabling SPARQL querying to capture and analyze distinctive stylistic patterns on the music surface. This process may provide valuable insights on the style, structure or emotional content of a music piece and can be used as part of feature engineering to machine learning applications.

Keywords

ontology engineering, semantic annotation, description logics, SPARQL, feature engineering, music harmony

1. Introduction

In music, the *pitch* is the perceptual interpretation of frequency [1]; it is associated to one of the 12 *pitch classes* A, A♯, B, C,..., G, G♯ and an *octave* (in which the pitch is located). A *note* (or tone) embodies a pitch and a duration; notes heard together constitute tone simultaneities (or *chords*). **Harmony** is concerned with the study of chords and their structure, progressions and relations. Tonal harmony in particular considers chords in the framework of **tonality**, the hierarchical organization of pitches around a *tonal center* (the tonic). *Functional (tonal) harmony* is a widely established and extensively studied harmonic system wherein chords assume specific functions depending on their relation to the tonic.[1] *Harmonic analysis* represents a multi-faceted musical task involving not only identifying chord roots and labeling chords but also segmentation into chords, defining points of key change, identifying non-chord tones and voice leading etc. Automated harmonic analysis has been a prominent area of research in computational

[1]A rough summary of functional harmonic precepts may be found in Appendix A.

musicology. For instance, [2] and [3] propose a chord model, a syntax for chord annotation and an algorithm to compute the chord root of a note simultaneity. Pardo and Birmingham [4] present an algorithm to segment the musical surface into chordal entities. *HarmTrace* [5] performs functional harmonic analysis on a sequence of annotated chords. In the 1990s, [6] and [7] have proposed systems for producing synthetic chorales in the style of J. S. Bach. Much more recently, [8] and [9] have used deep learning techniques for chorale generation.

The use of ontologies has been introduced in recent years to represent the music domain; a pioneer is *Music Ontology* [10], encompassing a wide range of concepts. *Music Note* [11] and *MusicNote* [12] propose a generic approach to representing, annotating, extracting, linking and searching music notation content. *Music Theory* ontology [13] captures the basic network of music theoretic concepts including Note, Interval, Chord and Progression (of notes or chords). *OMRAS2 Chord* ontology [14] provides a versatile vocabulary for describing chords and chord sequences. *Functional Harmony* ontology [15] proposes a thorough taxonomy of chords in the framework of functional harmony. *MusicPatternOWL* [16] and *Music Score* ontology (MusicOWL) [17] provide a comprehensive vocabulary for annotating music scores based on semantic information such as patterns, melodic fragments, dynamics and tonalities.

Chapter 2 illustrates the basic harmonic concepts in Hindemith's *Craft of Musical Composition* (CofMC)[2]. Chapter 3 introduces the *ontocomc* ontology with its main classes, properties, rules and extended functionality through the use of *Olwready2*. Chapter 4 discusses some basic queries and limitations relating to automated chord annotating as performed via *ontocomc*. Chapter 5 applies the ontology on Bach chorales annotating chords, graphically producing harmonic fluctuation and capturing harmonic patterns. Chapter 6 concludes the paper, summarizing main points and pointing to several directions for future research.

2. Hindemith's harmony in *The Craft of Musical Composition*

In this paper we distance ourselves from tonal harmony, following instead an alternative harmonic theory proposed in Paul Hindemith's **The Craft of Musical Composition** (1937). Hindemith considers the interval, i.e. the distance between two pitch-classes, as his primitive building block. **Intervals** are associated to *interval types* (major 2nd, perfect 4th, tritone etc.), which are ordered by diminishing strength in his *Series 2* (cf. Picture 7); Hindemith assigns an *interval root* to every interval (tritones excepted). **Chords** are viewed as sets of *intra-chord* intervals (i.e. intervals formed between two chord tones) and granted a chord root (or *degree*), namely the root of the strongest (by *Series 2*) of its intra-chord intervals; in the case of chords containing some tritone, they are also assigned a *guide-tone*. Just as intervals are classified according to their interval type, chords form an exhaustive **chord taxonomy**, in which every chord is assigned to a specific *subgroup*[3] by examining the nature of its intra-chord intervals. Dynamics in a chord sequence are governed by the fluctuation caused by the juxtaposition of simultaneities belonging to diverse subgroups as well as the progression of corresponding

---

[2]The reader is urged to brush over a sketchy survey of CofMC in Appendix B or, better still, consult [18].

[3]These six subgroups are indexed with Roman numerals I-VI. Chords in Subgroups V and VI do not have a root in principle, but may be assigned one depending on their context. In this paper, such chords are dubbed 'roving', in a rather arbitrary sense referring to Schoenberg in [19] and [20].

chord degrees. This is a highly original harmonic theory, at odds with any other known system, tonal or modal.

3. The *ontocomc* ontology

Hindemith's exhaustive chord taxonomy in CofMC (cf. Appendix **??**) lends itself quite naturally to a knowledge representation scheme on note simultaneities, enriching them with features that determine their inherent dynamics. In this paper, we propose a chord ontology called *ontocomc*[4], described in OWL language and developed using the Protégé 5.5.0 editor[5] [21]. At the highest level, *ontocomc* is anchored on four basic music objects represented with classes *PitchClass*, *Interval*, *Chord* and *Sequence*.

Pitch classes $A, B\flat$ (or its enharmonic equivalent $A\sharp), B, C, D\flat(C\sharp), D, E\flat(D\sharp), E, F, G\flat$ $(F\sharp), G, A\flat(G\sharp)$ are modeled via concept *PitchClass* and its 12 subclasses (*ontocomc.A*, *ontocomc.Bb* etc.). Class *ontocomc:A* contains individuals *ontocomc:a*, *ontocomc:gss* (for G$\sharp\sharp$), *ontocomc:ghh*[6] and *ontocomc:bbb* (for B$\flat\flat$), corresponding to enharmonically equivalent spellings used in common music practice; remaining interval subclasses are populated accordingly. **Pitches** are represented with the *Pitch* concept. In this paper, we consider pitches within the standard piano keyboard range, equivalent to MIDI numbers 21-108 (A0 to C8). Every *Pitch* individual gets related to (i) a *PitchClass* filler via object property *ontocomc:hasPitchClass* and (ii) an integer in [0,8] via datatype property *ontocomc:octave*.

The ontology defines **intervals** (Eq. 1) and classifies them into disjoint subclasses *NormIntv* and *Tritone*. Class *UndefinedInterval* is reserved for intervals with at least one note unspecified. *NormIntv* is partitioned into disjoint subclasses *RootLow* and *RootHigh* and their disjoint subclasses[7], as in Eq. 2:

$$Interval \equiv \exists high.PitchClass \sqcap \exists low.PitchClass$$
$$Interval \sqsubseteq NormIntv \sqcup Tritone \sqcup UndefinedInterval$$
$$NormIntv \sqsubseteq RootHigh \sqcup RootLow \tag{1}$$
$$UndefinedInterval \sqsubseteq \exists high.PitchClassUndef \sqcup \exists low.PitchClassUndef$$

$$RootHigh \sqsubseteq Maj2nd \sqcup Min2nd \sqcup Maj6th \sqcup Min6th \sqcup Perf4th$$
$$RootLow \sqsubseteq Maj3rd \sqcup Min3rd \sqcup Maj7th \sqcup Min7th \sqcup \tag{2}$$
$$\sqcup Perf5th \sqcup Octave \sqcup Unison$$

$$Maj2nd \sqsubseteq Maj2ndA \sqcup Maj2ndBb \sqcup Maj2ndB \sqcup ... \sqcup Maj2ndG \sqcup Maj2ndAb$$
$$Perf4th \sqsubseteq Perf4thA \sqcup Perf4thBb \sqcup ... \sqcup Perf4thAb \tag{3}$$
$$... \ ... \ ... \ ... \ ...$$

---

[4]http://purl.org/than.apos/ontocomc

[5]http://protege.stanford.edu

[6]The standard suffix for $\sharp$ is letter *s*. This may be confusing to some readers, as German nomenclature normally reserves *s* for $\flat$ (for example, *As* denotes A$\flat$), therefore letter *h* is here proposed as an alternative suffix for $\sharp$.

[7]Hindemith makes absolutely no distinction between enharmonic tones, which he considers as mere alternative spellings: in his view for example, diminished 4th A–Db is treated in exactly the same way as major 3rd A–C$\sharp$. In this sense, the taxonomy of intervals displayed in Eq. 2 is exhaustive.

$$Maj2ndA \equiv \exists high.B \sqcap \exists low.A$$
$$Maj2ndBb \equiv \exists high.C \sqcap \exists low.Bb$$
$$... \; ... \; ... \; ... \; ...$$
$$TritoneA \equiv \exists high.Eb \sqcap \exists low.A \tag{4}$$

As an example, subclass *Maj2nd* of *RootHigh* is further partitioned into 12 subclasses: *Maj2ndA*, *Maj2ndBb* and so on (Eq. 3). In particular, *Maj2ndA* corresponds to major seconds A-B and is explicitly defined (Eq. 4); similarly, *Maj2ndBb* corresponds to intervals Bb-C etc. The same rationale applies to all other subclasses of *NormIntv* as well as class *Tritone*.

At the highest level of this chord taxonomy (illustrated in Picture 1), chords are partitioned into two disjoint subclasses: *GroupA*, for chords containing no tritones and *GroupB*, for chords containing at least one tritone.

SubgroupV models augmented triads, while *SubgroupIII* corresponds to chords with at least one (major or minor) 2nd or 7th among their intra-chord intervals. Finally, *SubgroupI* is defined by means of a closure axiom on *GroupA*. These definitions are displayed in (5):

$$SubgroupV \equiv GroupA \sqcap \forall hasInterval.(Maj3rd \sqcup Min6th \sqcup UndefinedInterval)$$
$$SubgroupIII \equiv GroupA \sqcap \exists hasInterval.(Maj2nd \sqcup Maj7th \sqcup Min2nd \sqcup Min7th)$$
$$SubgroupI \equiv GroupA \sqcap (\neg(SubgroupIII \sqcup SubgroupV)) \tag{5}$$

GroupB chords are defined in an analogous way. In particular, *SubgroupII* is further partitioned into *SubgroupIIa* and *SubgroupIIb*; the former is defined as in (6):

$$SubgroupIIa \equiv SubgroupII \sqcap \; \leq 0hasInterval.Maj2nd$$
$$\sqcap \; \leq 1hasInterval.Tritone \sqcap RootPositionChord \tag{6}$$

Remaining chords get classified in *SubgroupIIb* and its subclasses.

The *ontocomc* ontology models the **chord sequence** as an independent entity; chords get linked to it as constituent parts [22], as shown in Figure 2. In essence, an *ontocomc:ChordSequence* individual provides a handle to the sequence object by linking to the initial chord via object property *ontocomc:firstChord*. Each chord is related to this individual via object property *ontocomc:constituentChord*. In its turn, a chord sequence generates its corresponding sequences of chord degrees, guide-tones and the two-voice framework[8].

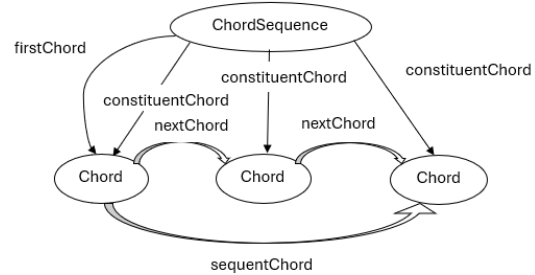**Object properties** *ontocomc:hasNote* with functional sub-properties *ontocomc:noteX* (for

Figure 2: The Sequence model used in *ontocomc*

---

[8]In CofMC, the sequence of intervals between the bass and the next most salient voice, typically the soprano.
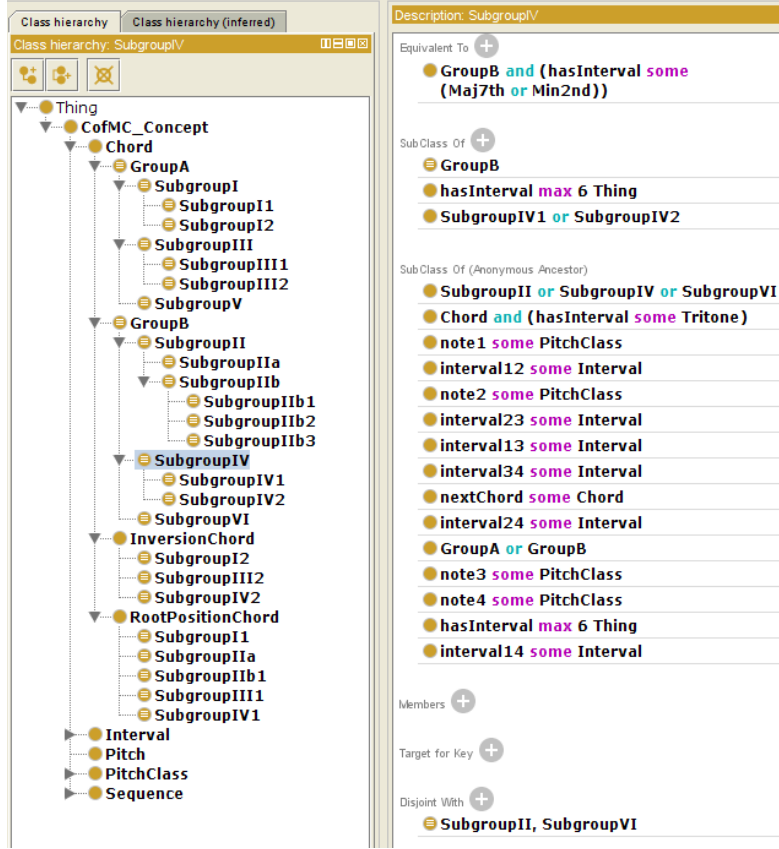
Figure 1: Chord taxonomy and example of a subgroup definition (*SubgroupIV*) in *ontocomc*

$X = 1, 2, 3, 4$) refer to the four constituent pitch-classes of a chord, while *ontocomc:hasInterval* with functional sub-roles *ontocomc:intervalXY* (for $X, Y = 1, 2, 3, 4 \,|\, X < Y$) denote the six intra-chord intervals. Object properties *ontocomc:low* and *ontocomc:high* relate a *Chord* instance to its two constituent *PitchClass* fillers. A chord is connected to all its subsequent ones in the sequence via object property *ontocomc:sequentChord*. Functional sub-property *ontocomc:nextChord* links the chord to the one immediately following; the last chord in line is typically 'grounded' by asseerting the triple *<last_chord> ontocomc:nextChord ontocomc:chordNil.* Role chain axiom

$$ontocomc : sequentChord \circ ontocomc : nextChord$$
$$SubPropertyOf \; ontocomc : sequentChord \tag{7}$$

enables reaching out iteratively to all subsequent chords in the sequence. Object properties *:degree, :frame, :guideTone, :nextDegree, :nextGuideTone, :nextFrame, :salientDissonance* and *:salientDissonanceResolution* pertain to **chord features** inferred during the chord analysis process. In particular, functional role *ontocomc:salientDissonance* denotes the most prominent dissonance[9]

---

[9]The term *dissonance* is here borrowed from tonal harmony and denotes seconds, sevenths or tritones.

(if any) among intra-chord intervals; object property *ontocomc:salientDissonanceResolution* naturally refers to its resolution in the following chord.

The ontology contains a set of SWRL rules, e.g. for identifying interval roots (Eq. 8):

$$RootHigh(?int), high(?int, ?note) \rightarrow intervalRoot(?int, ?note) \tag{8}$$

$$degree(?ch, ?n1), note1(?ch, ?n2), SameAs(?n1, ?n2) \rightarrow RootPositionChord(?ch) \tag{9}$$

Eq. 9 effectively 'pushes' a chord already in Subgroup III and with its root in the bass further down the taxonomy and into Subgroup III_1, due to equivalence axiom (10) in the TBox:[10]

$$SubgroupIII_1 \equiv SubgroupIII \sqcap RootPositionChord \tag{10}$$

Although we are by now able to produce most facts pertaining to a chord sequence, identifying degrees and guide-tones still eludes the expressivity supplied by the afore-mentioned DL axioms and SWRL rules; in this case, a pure algorithmic approach is needed. **Owlready2** is a Python module allowing 'ontology-oriented programming', i.e. object-oriented programming where the entities of an ontology may be treated as Python objects [24]. *Owlready* allows extending ontology classes with Python methods, in our case notably in order to compare intervals and identify the 'best' among them, using our familiar *Series-2* as a yardstick (cf. Appendix C). Methods extending class *Chord* may be applied in tandem to an entire chord sequence. In this way, degrees, guide-tones and salient dissonances in a chord sequence are explicitly computed in a Python/Owlready environment and entered at the same time as new facts in the ABox, while all remaining chord features (e.g. the frame, next degree, next guide-tone, next frame, salient dissonance resolution) as well as each chord's specific location in the chord taxonomy are implicitly inferred during the process.

4. Use cases and limitations

Tonal harmony has been thoroughly explored for well over 200 years and several systems ([5], [15]) for automated analysis have also been proposed. The originality of our system lies in the implementation of an alternative automated semantic annotation scheme for chords and chord sequences following Hindemith and CofMC. The *ontocomc* ontology provides a univocal and unambiguous characterization of a chord and its dynamics in a chord sequence.

The core ontology signature consists of 605 non-logical axioms, 225 classes, 41 object and 13 datatype properties. The ontology expressivity is $\mathcal{ALCOIQ(D)}$. In the following, we elaborate on some typical queries the ontology is able to handle:

- An **interval root** is inferred by matching the interval against the 12 interval types, classifying it as *RootHigh* or *RootLow* (or *Tritone*) and activating an SWRL rule as in Eq. 8. The interval must first be instantiated by having its low and high notes asserted in the ontology.

---

[10]All rules in *ontocomc* pertain to worlds locally closed *before* running the reasoner (i.e. the 4 notes of a chord, the 6 intra-chord intervals, the constituent chords of a given sequence); these rules are thus practically DL-safe [23] and are therefore expected to preserve decidability.

- The ontology classifies a chord in its specific **subgroup**, by comparing intra-chord intervals and pushing it down the chord taxonomy accordingly. The chord must first be instantiated by asserting its 4 notes and entering 6 fresh *Interval* individuals to represent the intra-chord intervals; asserting their low and high notes is inferred automatically.
- The ontology implements a full automated **harmonic analysis** of a chord sequence. Each chord is annotated with its distinct features (subgroup, root and guide-tone, salient dissonance and its resolution, chords following in the sequence etc.). The sequence can be loaded in *ontocomc* (e.g. from a csv file) by enriching the ABox with assertions about notes and intra-chord intervals for all chords and *nextChord* role assertions linking each chord to the next one in line. This pre-processing step can be implemented automatedly by means of a *Python/Owlready* script, as described in Chapter 5.
- Once a large repository of sequences and chords has been created, the space can be searched for **distinctive stylistic patterns** satisfying specific sets of features, i.e. chord degree progressions, cadential figures etc.

At its present edition, the ontology does not support tonal center analysis. In a nutshell, Hindemith considers the chord degree sequence as a field of interrelations governed by his *Series 1*; whenever dynamics are altered, a modulation ensues. The integration of this aspect is a major topic in ongoing research.

In principle, analogies with existing ontological representations based on tonal/modal harmony (cf. Chapter 1) are far from trivial, due to the large gap between the two worldviews. Some primitive correspondences do exist though, for instance triads practically map to Subgroup I chords and seventh chords generally belong to Subgroup II. More elaborate and insightful connections to existing music ontologies should allow getting the best of both worlds.

5. Application to the Bach chorale corpus

A chorale is a hymn of the Lutheran church, its words and melody intended to be sung by the whole congregation. Soon after its becoming part of the church service, it grew customary to supply the melody with a harmonic accompaniment, a practice reaching its peak with late-baroque masters. The melody is typically entrusted to the highest voice and combined with a counterpoint in 3 additional lines. The chorales of Johann Sebastian Bach (1685-1750) are considered perfect artifacts imbued with a thorough understanding of the dynamics of tonality; the three lower voices shape the harmonic surface and allow for seamless transitions between tonal regions. Bach achieves a perfect balance between the horizontal and the vertical element through the deft and often bold use of **non-chord** (or nonessential) tones, empowering individual lines with a melodic content of their own without ever compromising tonal unity.

The chord annotation and analysis procedure proposed in this paper works in two steps. At the **chord sequence generation step**, a *.mid* music file[11] is received at the input and the

---

[11]The entire Bach chorale corpus is uploaded at [25] in standard MIDI file format. A MIDI file is arguably the most widespread method for symbolic music representation. It is a list of music events, notably those represented by the note-on and note-off messages. Each such message is equipped with a MIDI note number encoding the pitch, a key velocity value for intensity, a channel specification and a timestamp.[26]

Figure 3: J. S. Bach's chorale 'Herzliebster Jesu' BWV 244.46

information is processed to produce 4 arrays of events, corresponding to the four voices. These *horizontal* sequences are then aligned over time to produce one single sequence of *vertical* chordal formations. Tokenization of tone simultaneities is carried out at quaver note[12] time resolution (the equivalent of two MIDI-file timestamps differing by 512 time units). As onset times and token timestamps do not always coincide, a suitable *unfolding* must be performed (a similar procedure is used in [27]), wherein notes are duplicated at every quaver (and tagged as *tied*, if there is no new onset at that particular metric location). Once the chord sequence has been thus generated, the ontological space is populated with chord instances (cf. Appendix D); this is followed by the **harmonic analysis step** implementing chord annotation.

As an example monitoring the entire process, we introduce J. S. Bach's chorale *'Herzliebster Jesu'* from the St-Matthew Passion (Figure 3)[13]. At first, the chord sequence of the opening phrase is generated and tokens (chords) instantiated by having *ontocomc:hasNote* and *ontocomc:nextChord* object properties suitably filled in. Chord names are indexed after quaver numbers, odd (even) numbers corresponding to (non-) accentuated quavers. In other words, chords are labeled as *<sequence name>_<quaver number>*, thus the first element of sequence *ontocomc:herzliebster1* is labeled *ontocomc:herzliebster1_7* (note the incomplete first bar at Figure 3). Chord analysis is then carried out in Python/Owlready; results are shown in Figure 4[14].

With core ontology populated and enriched with chord attributes, we can graphically set out harmonic fluctuation; that corresponding to the first and second phrases of *Herzliebster Jesu* is shown in Figures 5 and 6, respectively. A conventional full-fledged harmonic analysis

---

[12]A quaver or eighth note is a musical note (or pause) played for one eighth the duration of a whole note (semibreve). It is therefore a time unit relative to other rhythmic values. Its duration is half that of a crotchet (or quarter note) and twice that of a semiquaver (or sixteenth). Quavers in Bach chorales are typically used for voice leading, while the use of semiquavers is rather sporadic and generally reserved to ornamentation. The choice of quaver (over semiquaver) quantization keeps the ontology reasonably uncluttered, missing little information in the process.

[13]downloaded from *https://www.bach−chorales.com* by Luke Dahn

[14]The reasoner needs to be called twice during execution of Python scripts: once to classify intra-chord intervals for the whole sequence in one single batch and a second time during the *derove* process to classify fresh intervals between 'roving' chord tones and neighboring degrees.

```
herzliebster1_7: [ontocomc.SubgroupIIb2, ontocomc.InversionChord] (4, ontocomc.b) (1, ontocomc.eb)
herzliebster1_8: [ontocomc.SubgroupIIb2, ontocomc.InversionChord] (4, ontocomc.b) (1, ontocomc.eb)
herzliebster1_9: [ontocomc.SubgroupI1] (1, ontocomc.e) (1, ontocomc.e)
herzliebster1_10: [ontocomc.SubgroupI1] (1, ontocomc.e) (1, ontocomc.e)
herzliebster1_11: [ontocomc.SubgroupVI, ontocomc.InversionChord] (4, ontocomc.b) (0, ontocomc.pitchClassUndef)
herzliebster1_12: [ontocomc.SubgroupVI, ontocomc.InversionChord] (4, ontocomc.b) (0, ontocomc.pitchClassUndef)
herzliebster1_13: [ontocomc.SubgroupI1] (1, ontocomc.gb) (1, ontocomc.gb)
herzliebster1_14: [ontocomc.SubgroupI1] (1, ontocomc.gb) (1, ontocomc.gb)
herzliebster1_15: [ontocomc.SubgroupI1] (1, ontocomc.gb) (1, ontocomc.gb)
herzliebster1_16: [ontocomc.SubgroupI1] (1, ontocomc.gb) (1, ontocomc.gb)
herzliebster1_17: [ontocomc.SubgroupI2] (2, ontocomc.e) (2, ontocomc.e)
herzliebster1_18: [ontocomc.SubgroupI2] (2, ontocomc.e) (2, ontocomc.e)
herzliebster1_19: [ontocomc.SubgroupIIb2, ontocomc.InversionChord] (3, ontocomc.gb) (1, ontocomc.bb)
herzliebster1_20: [ontocomc.SubgroupIIb2, ontocomc.InversionChord] (3, ontocomc.gb) (1, ontocomc.bb)
herzliebster1_21: [ontocomc.SubgroupI1] (1, ontocomc.b) (1, ontocomc.b)
herzliebster1_22: [ontocomc.SubgroupIII1] (1, ontocomc.b) (1, ontocomc.b)
herzliebster1_23: [ontocomc.SubgroupI1] (1, ontocomc.b) (1, ontocomc.b)
herzliebster1_24: [ontocomc.SubgroupI1] (1, ontocomc.b) (1, ontocomc.b)
herzliebster1_25: [ontocomc.SubgroupVI, ontocomc.InversionChord] (4, ontocomc.e) (0, ontocomc.pitchClassUndef)
herzliebster1_26: [ontocomc.SubgroupVI, ontocomc.InversionChord] (4, ontocomc.e) (0, ontocomc.pitchClassUndef)
herzliebster1_27: [ontocomc.SubgroupI1] (1, ontocomc.b) (1, ontocomc.b)
herzliebster1_28: [ontocomc.SubgroupI1] (1, ontocomc.b) (1, ontocomc.b)
herzliebster1_29: [ontocomc.SubgroupI1] (1, ontocomc.gb) (1, ontocomc.gb)
herzliebster1_30: [ontocomc.SubgroupI1] (1, ontocomc.gb) (1, ontocomc.gb)
```

Figure 4: *Herzliebster Jesu*, phrase 1, chord analysis. Each chord corresponds to a quaver slice. The output displays (a) chord classification, (b) the degree's pitch-class and position in the chord and (c) same info for guide-tones. Subgroup VI ('roving') chords have been granted a root, though guide-tones are left undefined.

of the second phrase, distinguishing non-chord tones from actual tonal harmonies[15], yields the green-line curve in Figure 6; harmonic tempo normally proceeds by *crotchets*, therefore contrapuntal voice-leading in *quavers* has largely been suppressed[16]. Harmonic tempo slows further down to a *minim* at quaver 49: a single A7 chord dominates the whole first half of bar 7 leading to the fermata over a D-major triad, practically brushing off all intermediate simultaneities as mere off-springs due to voice-leading.

The blue-line curve in Figure 6 illustrates a different approach. Here, time quantization has been (doggedly) kept at a strict quaver, allowing us to register all time simultaneities occurring at this pace. This is admittedly not a 'proper' harmonic analysis; nevertheless, it sheds light in several ways into the workings of harmony in general and Bach's style in particular. Rather than dismissing voice-leading, we allow the full effect of non-chord tones to move to the forefront. Notwithstanding their transitory nature, non-chord simultaneities induce a momentary **harmonic tension** in musical texture which largely remains under the carpet

---

[15]In this case, the segmentation to actual harmonies has been done manually.

[16]The second half of bar 6 is harmonically ambiguous. The last beat (at 47-48) consists of two minor seventh chords, neither of which would normally be expected as a self-contained entity in Bach's times and style. The whole second half of the bar could be treated as a single B minor triad with extended voice leading in all 4 voices. A secondary dominant on E is nevertheless clearly implied at the very last quaver. This is an example illustrating the challenges of tonal harmony, where the musical surface is prone to multiple conflicting analyses. Instead, Hindemith's and our approach inherently leads to a univocal chord annotation.
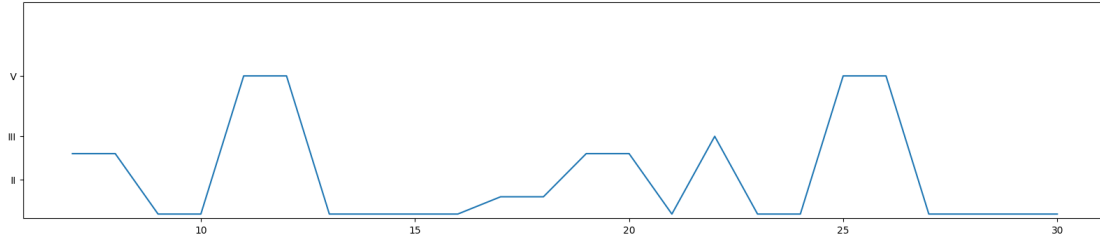
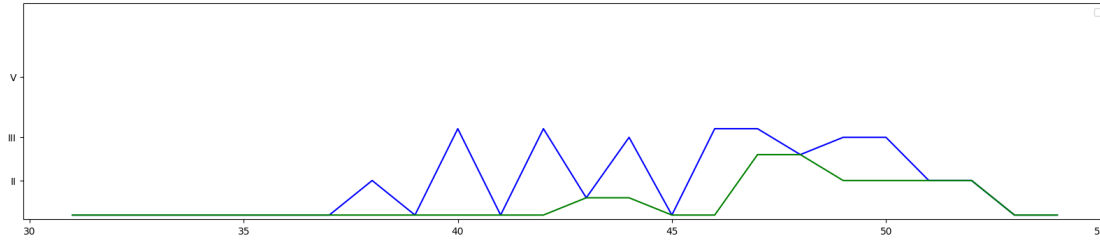**Figure 5:** *Herzliebster Jesu*, phrase 1, harmonic fluctuation

Figure 6: *Herzliebster Jesu*, phrase 2, harmonic fluctuation

Q1	*Chord and beat value "off" and (SubgroupIII or SubgroupIV)* *and inverse(nextChord) some (Chord and (SubgroupI or SubgroupII))*
A1	$herzliebster2\_38, herzliebster2\_40, herzliebster2\_42, herzliebster2\_44, herzliebster2\_46$
Q2	*Chord and beat value "on" and (SubgroupIII or SubgroupIV)* *and nextChord some (Chord and (SubgroupI or SubgroupII))*
A2	$herzliebster2\_47$

Table 1

DL-querying after voice-leading involving non-chord tones in *Herzliebster Jesu*, phrase 2. Q1 singles out non-tertian simultaneities with passing/alternating tones, typically occurring *off* the beat. Q2 identifies a single non-tertian simultaneity with an appoggiatura, typically occurring *on* the beat.

and escapes quantification by traditional analytical procedures. Back to our example, the first phrase of *Herzliebster Jesu* is largely composed in block chords (mainly triads and dominant sevenths, corresponding to the low hills of Subgroup I/II chords in Figure 5) and produces a rather smooth curve, notably excepting the high plateaus due to two diminished chords. The ragged blue-line curve of the second phrase highlights a sharp contrast due to the constant injection of passing dissonances and a longer state of high tension at quavers 46-50.

Composers use voice-leading to shape the music surface, a procedure which causes harmonic tension to fluctuate. The *ontocomc* ontology permits us to monitor the latter and, through this, acquire a tool to analyze the former, in particular visualize and capture specific voice-leading patterns. In principle, non-chord tones cause the formation of non-tertian chords which typically end up as members of subgroups III and IV in our system. Non-chord tones (such as passing/alternating tones or appoggiaturas) may be captured e.g. in the second phrase of our chorale, even by simple DL queries, as illustrated in Table 1. SPARQL queries enable a closer

SubgroupIII_offbeat	prev_deg	deg	next_deg
herzliebster2_38	d	d	d
herzliebster2_40	g	b	e
herzliebster2_42	e	b	e
herzliebster2_44	e	a	b
herzliebster2_46	b	e	b

Table 2
Herzliebster Jesu, phrase 2. The table displays off-beat *Subgroup III* chords, flanked on both sides by on-beat tonal chords; respective 3-grams highlight the injection of non-chord quavers on unaccented beats.

scrutiny on the use of non-essential notes; Table 2, for example, displays the result of a query about progressions of 3 chords (3-grams), starting and ending on accented beats; the middle one is a (passing) *Subgroup III* chord, flanked on both sides by tonal triads or seventh chords (roughly corresponding to *Subgroups I* and *II*). A detailed analysis on such 3-grams (starting both *on* and *off* the beat) can be applied to the entire repository of Bach chorales, suitably stored on *GraphDB*[17] for example, in order to provide insights on voice-leading as a stylistic feature. An entire set of SPARQL queries can thus be used to explore several aspects of Bach's distinctive style.

6. Conclusion and future work

In this paper, we have proposed a novel system implementing automated semantic annotation for chords based on Hindemith's *Craft of Musical Composition*, deploying the representational powers of ontologies. We have introduced the *ontocomc* ontology and have used it as a tool to analyze a Bach chorale, populating the ontology with chord instances and features. We have also briefly shown how *ontocomc* may be used to search for distinctive chord patterns via DL and SPARQL queries.

As we have seen in Chapter 5, the *ontocomc* ontology is a valuable tool to analyze harmonic tension, which deserves further exploring in future research. Harmonic fluctuation and its visualization may provide music education with a useful tool illustrating the voice-leading mechanism and could be proposed as part of the curriculum for harmony courses. Another area of research consists in exploring Bach's style (e.g. see [28] and [29] for patterns to match against the entire Bach chorale corpus properly annotated and stored in a repository). As mentioned in Chapter 4, expanding *ontocomc* in order to integrate tonal center analysis is part of ongoing research, aiming at identifying the exact locations of modulations on the musical surface. Finally, enriching chord instances with respective features can be integrated as part of learning systems towards music generation applications.

---

[17]https://graphdb.ontotext.com/documentation/11.0/index.html

References

[1] A. Anglade, Logic-based Modelling of Musical Harmony for Automatic Characterisation and Classification, Ph.D. thesis, School of Electronic Engineering and Computer Science, Queen Mary, University of London, 2014.

[2] C. Harte, M. Sandler, S. Abdallah, E. Gómez, Symbolic representation of musical chords: a proposed syntax for text annotation, 16th International Conference on Music Information Retrieval, Proceedings (ISMIR, 2005).

[3] E. Cambouropoulos, M. Kaliakatsos-Papakostas, C. Tsougras, An Idiom-independent Representation of Chords for Computational Music Analysis and Generation, Joint 40th International Computer Music Conference (ICMC) and 11th Sound and Music Computing (SMC) Conference (2014).

[4] B. Pardo, W. P. Birmingham, The Chordal Analysis of Tonal Music, Technical Report, Electrical Engineering and Computer Science Department, University of Michigan, 2001.

[5] W. B. de Haas, J. P. Magalhães, R. C. Veltkamp, F. Wiering, HarmTrace: Improving Harmonic Similarity Estimation Using Functional Harmony Analysis, 12th International Society for Music Information Retrieval (ISMIR) Conference (2011).

[6] K. Ebcioğlu, An Expert System for Harmonizing Chorales in the Style of J. S. Bach, The Journal of Logic Programming, Elsevier (1990).

[7] H. Hild, J. Feulner, W. Menzel, Harmonet: A Neural Net for Harmonizing Chorales in the Style of J. S. Bach, in: J. E. Moody, S. J. Hanson, R. P. Lippmann (Eds.), Proceedings of the 5th International Conference on Neural Information Processing Systems (NIPS), Morgan Kaufmann Publishers Inc., 1991, pp. 267–274.

[8] G. Hadjeres, F. Pachet, F. Nielsen, DeepBach: a Steerable Model for Bach Chorales Generation, in: D. Precup, Y. W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, 70, JMLR, Sidney, Australia, 2017.

[9] F. Liang, M. Gotham, M. Johnson, Automatic Stylistic Composition of Bach Chorales with Deep LSTM, in: S. J. Cunningham, Z. Duan, X. Hu, D. Turnbull (Eds.), Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR, Suzhou, China, 2017.

[10] Y. Raimond, S. Abdallah, M. Sandler, F. Giasson, The Music Ontology, in: Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR, Vienna, Austria, 2007, pp. 417–422.

[11] A. Poltronieri, A. Gangemi, The Music Note Ontology, in: K. Hammar, C. Shimizu, H. K. McGinty, L. Asprino, V. A. Carriero (Eds.), Proceedings of the 12th Workshop on Ontology Design and Patterns (WOP 2021), CEUR-WS, 2021.

[12] S. S. Cherfi, C. Guillotel, F. Hamdi, P. Rigaux, N. Travers, Ontology-based Annotation of Music Scores, in: The Knowledge Capture Conference, Association for Computing Machinery, Austin, Texas, USA, 2017.

[13] S. M. Rachid, D. D. Roure, D. L. McGuinness, A Music Theory Ontology, in: Proceedings of the 1st International Workshop on Semantic Applications for Audio and Music, Association for Computing Machinery, Monterey, California, USA, 2018.

[14] C. Sutton, Y. Raimond, M. Mauch, The OMRAS2 Chord Ontology, 2007. URL: http://purl.org/ontology/chord.

[15] S. Kantarelis, E. Dervakos, N. Kotsani, G. Stamou, Functional Harmony Ontology: Musical Harmony Analysis with Description Logics, in: Journal of Web Semantics, 75, Elsevier, 2023.

[16] C. E. Achkar, T. Atéchian, Supporting Music Pattern Retrieval and Analysis: an Ontology-based Approach, in: Proceedings of the 10th International Conference on Web Intelligence, Mining and Semantics, Association for Computing Machinery, Biarritz, France, 2020.

[17] J. Jones, D. de Siqueira Braga, K. Tertuliano, T. Kauppinen, MusicOWL: the Music Score Ontology, in: International Conference on Web Intelligence, Association for Computing Machinery, Leipzig, Germany, 2017.

[18] P. Hindemith, Craft of Musical Composition, Book I, Associcated Music Publishers Inc., 1937.

[19] A. Schoenberg, Harmonielehre, Universal Edition, 1911/22.

[20] A. Schoenberg, Structural Functions in Harmony, Norton and Co., 1948.

[21] M. A. Musen, The Protégé project: A look back and a look forward, Association of Computing Machinery Specific Interest Group in Artificial Intelligence 1 (2015).

[22] N. Drummond, A. Rector, R. D. Stevens, G. Moulton, Putting OWL in Order: Patterns for Sequences in OWL, in: B. C. Grau, P. Hitzler, C. Shankey, E. Wallace (Eds.), Proceedings of the OWLED*06 Workshop on OWL: Experiences and Directions, CEUR-WS, Athens, Georgia, USA, 2006.

[23] M. Krötzsch, Description Logic Rules, Ph.D. thesis, Karlsruher Institut für Technologie, 2010.

[24] J.-B. Lamy, Ontologies with Python, Programming OWL 2.0 ontologies with Python and Owlready2, Apress, 2021.

[25] L. Dahn, The four-part chorales of J. S. Bach, 2021. URL: https://kunstderfuge.com.

[26] The MIDI Association, ???? URL: https://www.midi.org/specifications-old/item/the-midi-1-0-specification.

[27] L. Bigo, D. Conklin, Four-part Chorale Transformation by Harmonic Template Voicing, Research Report, Universidad del País Vasco (2016).

[28] C. Koechlin, Étude sur les Notes des Passage, Max Eschig, 1922.

[29] C. Koechlin, Étude sur le Choral d' École, Eugel, 1929.

[30] H. Riemann, Harmony Simplified, the Theory of the Tonal Functions of Chords, Augener, London, 1896.

[31] W. C. Mickelsen, Hugo Riemann's Theory of Harmony, University of Nebraska Press, 1977.

[32] E. Aldwell, C. Schachter, Harmony and Voice Leading, 3rd ed., Shirmer, 2002.

[33] Y. Sadaï, Harmony in its Systemic and Phenomenological Aspects, Yanetz, 1980.

[34] P. Hindemith, A Concentrated Course in Traditional Harmony, Associated Music Publishers, 1943.

[35] J. R. Halliday, Paul Hindemith, the Theorist, Ph.D. thesis, Eastman School of Music, 1941.

[36] P. Hindemith, A Composer's World, Harvard University Press, 1953.

Appendix

A. A bird's-eye view of tonal functional harmony

Tonal harmony examines the structure of chords and their relations within a tonal framework, governed by the concept of key (or mode). A basic building block is the **interval** between two tones, in some sense their *distance*, as determined by their frequency ratio. The smallest such interval is the second, either a semitone or a whole tone, the latter consisting of two consecutive semitones. In general, intervals are denoted by the number of intermediate tones: the interval A-C, for instance, is a third (referring to the 3 first notes of the familiar 7-note series A, B, C, D, E, F and G). Thirds, sixths and sevenths are further classified as major or minor, according to the number of consecutive semitones they contain: back to our example, A-C is a minor third (consisting of 3 semitones in total, whereas a major third would comprise 4). Fourths and fifths are perfect (consisting of 5 and 7 semitones, respectively), augmented or diminished. Specifically, a diminished fifth is equivalent to an augmented fourth; they comprise 6 semitones (or 3 whole tones) each and are also called tritones. Intervals may either be **consonant** (minor and major thirds, perfect fourths and fifths, minor and major sixths, unisons and octaves) or **dissonant** (practically all the rest, i.e. minor and major seconds, minor and major sevenths and all diminished and augmented intervals without exception).[18]

Generally speaking, a piece of music composed in the tonal idiom is based on a **tonality**, usually a major or minor key, sometimes one of the older church modes (lydian, dorian etc.). Each such key or mode is a distinct sequence of 7 pitch-classes (out of 12 in total) in preset distances of one or two semitones between two consecutive ones. The first note is called the tonic or first degree, followed by the second, the third etc., each tagged by a latin numeral (I to VII).

In tonal harmony, **chords** are formed by stacking notes upon one another, typically at the distance of a third, following the **tertian principle**. A three-tone chord or triad consists therefore of two successive thirds, the outer tones thus lying a fifth apart. The lower note (the bass) is called the **root**, the other ones are named the third and the fifth of the chord.

Tonal chords are classified as

- Consonant, namely major (resp. minor) triads, in which the third lies a major (resp. minor) third above the root and the fifth of the chord always a perfect 5th apart
- Dissonant, i.e. any remaining triads and all chords with 4 or more (distinct) tones. Such simultaneities consist of pitch-classes perceived as somehow incompatible to each other and therefore tend towards some sort of reparation or **resolution** of the dissonance.

Stacking a fourth tone on top of a triad creates a seventh between the outer voices[19], producing a 7th-chord (inevitably a dissonant one). In case the root cedes its place at the bass to some other chord tone, the chord is denoted as inverted. **Inversions** are typically used to enrich harmonic

---

[18] There exist diminished and augmented intervals other than 4ths and 5ths, for example the diminished third A – C flat. These intervals are enharmonically equivalent to some 'simpler' interval (A – B, in our case) and normally occur with altered chords in tonal harmony.

[19] A chord sequence is typically viewed as a series of vertical simultaneities intended for a mixed chorus in 4 parts (soprano, contralto, tenor and bass, from the top voice downwards).

texture and allow for a smoother voice leading and an improved bass line. Hugo Riemann (1849-1919) introduced the concept of **functional harmony** in the late 19th century, describing two competing and counterbalancing harmonic forces at play [30, 31]: the **subdominant** SD, mainly represented by the major chord on the fourth degree and the **dominant** D, with the major chord on the fifth degree as its principal agent. The former normally tends to the latter and this in turn to the **tonic** T (the major chord on the first degree): the succession IV-V-I (SD-D-T) constitutes the **perfect cadence**, a harmonic schema firmly establishing the tonality [32]. Chords on I, IV and V are called primary; auxiliary chords on the remaining degrees typically perform one of the above functions as substitutes to primary chords.

The seven pitch-classes of a mode constitute the diatonic environment. By applying chromatic inflections to these 7 tones, one may obtain the remaining 5 pitch-classes and enrich the harmonic texture with altered chordal formations, secondary dominants etc. A much wider choice of chords expands tonal functions to whole regions across the musical surface. The general tendency is always from the subdominant to the dominant, although the route may become tortuous, gradually gaining momentum before finally landing on the tonic, in what may still be considered as a more or less extended cadence [33]. This course may also include key changes, either transient (*tonicization*) or more or less persistent (*modulation*). Modulations to closely-related keys (during the exposition section of a piece) or more remote ones (in the development section) give prominence to centrifugal tendencies, at the same time providing structure and variety to the musical surface. Transitions between keys are not always sharp and intermediate chords may be open to multiple tonal meanings, a state that Schoenberg [19] calls vagrant or *roving harmony*.[20] The dynamics of tonal harmony are regulated by two complementary forces. On the one hand, one has the *vertical* aspect of chords and their functions across keys and tonal regions. On the other hand, there is the *horizontal* element implemented by the simultaneous voice leading of several contrapuntal lines. This interplay between harmony and counterpoint gains impetus by the use of neighboring and passing tones, suspensions, appoggiaturas etc. These non-chord tones are injected on the musical surface in-between occurrences of actual (tertian) chords and are perceived as circumstantial and irrelevant to the overall harmonic scheme. Such notes create transitional vertical formations, typically not conforming to the tertian principle of chordal construction[21]. In principle, these note simultaneities have a dissonant nature and induce tension in the texture; suspensions and appoggiaturas normally occur on accented beats and the ensuing tension is rather high, while passing and neighboring tones usually occur off-beat and generally create a milder effect.

---

[20]Considering modulations in a musical piece not as mere arbitrary key changes but as a fine interplay between tonal regions leads to a more integrated view of tonality; Schoenberg pushes this idea to the limit with his principle of monotonality: "…every digression from the tonic is considered to be still within the tonality, whether directly or indirectly, closely or remotely related. In other words, there is only one tonality in a piece, and every segment formerly considered as another tonality is only a region, a harmonic contrast within that tonality" [20]

[21]Certain such formations are routinely used in idioms beyond classical music and have acquired an independent harmonic status, e.g. suspended or major 9th chords in jazz and rock music.

B. Paul Hindemith's *Craft of Musical Composition*

"Our old friend Harmony, once esteemed the indispensable and unsurpassable teaching method, has had to step down from the pedestal upon which general respect had placed her" [34]

In 1937, Paul Hindemith[22] published the first book (Theoretical Part) of his *Craft of Musical Composition* (CofMC) [18]. Two further volumes were completed in the 40's, dealing with the practical aspect of writing in 2 and 3 parts. A fourth book on four-part writing was to conclude the series but was left incomplete at the time of his death.

Hindemith starts by analyzing the overtone series to explore the nature of the twelve pitch-classes. Notes of the twelve-tone scale arise one by one through quite intricate elaborations on partials produced by a fundamental or generator tone. The order in which these are produced reflects their relative value in relationship to the fundamental. The first ones, namely those one fifth or one fourth away from the generator, bear a close relationship to it; as we progress further in the sequence the relationship grows weaker, until it becomes barely perceptible in the tone lying at the distance of a tritone. This is Hindemith's **Series 1**, ranking all 12 pitch-classes in order of their closeness to the generator tone (Figure 7) . In effect, this series replaces the concepts of key and tonality in tonal music.

Intervals enter next: Hindemith examines the simultaneous sounding of two tones, this time focusing on the phenomenon of combination tones[23]. As a result, he ranks all 12 possible intervals in **Series 2**: in this sequence, one moves from the (harmonically) steadier intervals of the fifth and the fourth to the unstable seconds and sevenths, by way of thirds and sixths. Intervals come in pairs, as can be seen in Figure 7[24]; inverted ones immediately follow their counterparts and are therefore slightly inferior to them. We now come to a concept of primal importance in the theory of CofMC: of the two tones, the one best related to the combination tones (produced by the simultaneous sounding) naturally stands out; Hindemith considers this as the fundamental and calls it the **interval root**. This is a wholly original concept and does not correspond to any notion in standard tonal theory. Of each pair of intervals in Series 2, the first has its root coinciding with the low note; its inverted counterpart has the same root, albeit this time located high, a fact which accounts for its relative instability (Figure 8). Series 1 and 2 are Hindemith's basic building blocks, the former providing the large connecting elements, the latter supplying the small building material. [35] The two series look almost similar at first sight, yet they embody two wholly different concepts. Series 1 assigns tonal positions to all 12

---

[22]Composer, viola soloist and conductor, born in 1895 near Frankfurt, a leading figure in 20th century music. Extremely prolific, he composed operas, symphonic music, concertos, chamber music etc. Abandoning Nazi Germany in the 30s, he emigrated to USA and taught at Yale and elsewhere. He returned to Europe after the war, continued composing and resumed teaching in Zurich, where he lived until his death in 1963. Apart from The *Craft of Musical Composition*, he has notably written *Traditional Harmony* (in 2 volumes) and *Elementary Training for Musicians*.

[23]Combination tones (or difference or Tartini tones) are barely perceptible bass frequencies when two tones sound together. A violinist may hear them as soft bass tones when she plays double stops perfectly in tune; organ manufacturers make use of them in small organs to produce a low tone with two smaller pipes in place of a large one. [35]

[24]Images in this section are reproduced from the Greek edition of CofMC with permission from publisher Stefanos Nasos.
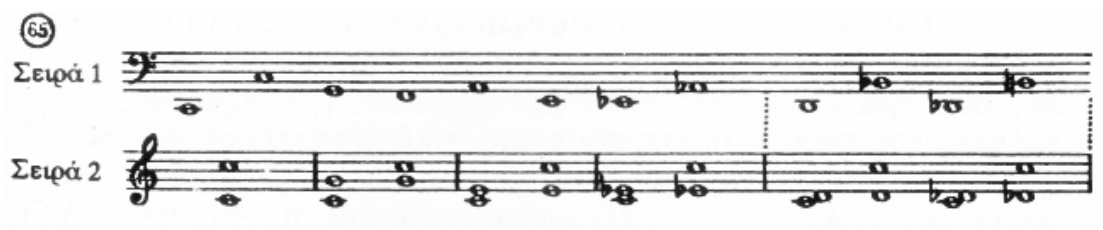
Figure 7: Series 1 and 2

Figure 8: Interval roots in *Series 2*. The image illustrates harmonic (above) and melodic potential (below the stave).

tones in relation to a given generator (the tonal center, assumed to be pitch-class C in Figure 7). Series 2 ranks *intervals* in diminishing harmonic potential while traversing the series from left to right; the melodic potential of intervals is determined by the same series, only this time in reversed direction (Figure 8). Thus, intervals bear an inherent amount of tension, in a sense a "load", non-decreasing as we move to the right. The octave and the unison sound perfectly transparent at one end. The tritone holds the other end: its potential is so high that it needs some kind of resolution. The tone closest to the root of the resolving interval acts as the **root representative** of the tritone, while the other tone is called the **guide-tone**.

Hindemith is a harsh critic of the traditional (tonal) harmonic theory. In his view, tertian harmony offers a much too restrictive view of the wealth and complexity of tone simultaneities occurring in contemporary music. In CofMC, a chord is defined as **any simultaneity whatsoever**, whether tertian or other. It is the dynamics between intra-chord[25] intervals which determine the nature of the chord. Interval roots fight to exert their influence on fellow intra-chord intervals; the root corresponding to the 'best' intra-chord interval (i.e. the one leftmost placed in Series 2) becomes the root (the **degree**) of the chord (cf. Figure 9).

Hindemith maintains that the nature and individual features of a chord are basically invariant to translations of its tones across octaves, provided their order is not altered. Moreover, an inversion[26] comes about not by shifting around the tones of a chord but simply by transplanting the root upward [35]. Indeed, chords with the root at the bass tend to be more stable than

---

[25]We introduce here this term to denote the intervals formed between two constituent tones of a chord.

[26]Hindemith avoids the term 'inversion' in general. In the following, we will be using terms 'root position' (to indicate the root lying in the bass) and 'inversion' (otherwise) for convenience, independently from their tonal homonyms.

Figure 9: Chords and their roots. The three upper staves display chords and their respective 'best' intervals and chord roots (from top to bottom). The lower stave illustrate more complex simultaneities; best intervals are marked with square brackets and roots with arrows.

formations consisting of the same tones but with their root at a higher voice and hence are somewhat better placed in the chord taxonomy. We have now hit at the core of CofMC's harmonic theory: Hindemith assigns all tone simultaneities, irrespective of their number of distinct tones or their complexity, to specific classes and subclasses. Chords basically classify into two large groups, according to the presence (or absence) of the tritone among intra-chord intervals. Chords with no tritones belong to **Group A**, while chords containing at least one tritone belong to **Group B**. Chords in the latter group lack the stability of those in the former, precisely because of the tritone gravitating towards its resolution, leading the entire chord along. In general, chords are classified into subgroups according to the nature of their intra-chord intervals. The two above-mentioned groups are further subdivided, Group A (Group B) into Subgroups I, III and V (Subgroups II, IV and VI). Further subdivisions pertain to chords in root position or in inversion. In this way, Hindemith constructs a complete hierarchy of tone simultaneities, subgroups at the finest level partitioning the space into pairwise disjoint classes.

  Subgroup I consists of major and minor triads, in root position ($Subgroup I_1$) or inversion ($Subgroup I_2$). **Subgroup II** consists of chords in which the tritone is dominated by some stronger interval. In particular, $Subgroup II_a$ contains our familiar dominant seventh chords in root position. The presence of a major second places the chord in *Subgroup II_b*, further divided into subgroups II_{b1}, II_{b2} and II_{b3}. The former two correspond to chords with one tritone, in root position and inversion respectively; the latter subgroup consists of chords with at least two tritones. Incidentally, *Subgroup II_{b2}* contains the inversions of dominant chords and similar constructions. **Subgroup III**, a 'rough and unpolished race' in Hindemith's colorful description, corresponds to a very large family, in essence made up of any tone simultaneity comprising seconds or sevenths or both (and obviously no tritone). A small subset of those
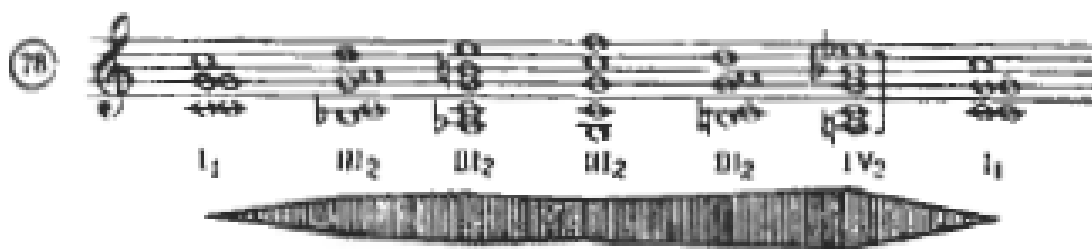
Figure 10: The interplay between degree progression, the two-voice framework and harmonic fluctuation

may be identified as the secondary triads with added 7th and their inversions. Again, chords in root position fall under *Subgroup III_1*, inversions under *Subgroup III_2*. **Subgroup IV**, a 'strange set of piquant, coarse and highly colored chords', contains tone simultaneities with minor seconds or major sevenths or both; tritones are again subordinate to stronger intervals. This subgroup is similarly divided into *Subgroups IV_1 and IV_2*. The last two subgroups are rather sparsely populated. They consist of simultaneities with no definite direction, open to multiple interpretations. As a result, no clear root can be assigned to them a priori, though this may change in the process, based on harmonic context; in other words, the same chord may be provided with a different root, depending on the chords surrounding it. **Subgroup V** is made up of Group-A chords consisting of nothing but major thirds and minor sixths, while **Subgroup VI** contains Group-B chords with no other intervals but tritones, minor thirds and major sixths[27]. The latter possess no intra-chord interval strong enough to dominate the tritone, which imbues therefore the whole chord with its inherent vagueness. Referring to Group-B chords in general, the tone which belongs to some tritone and at the same time forms the strongest intra-chord interval with the root assumes the role of guide-tone[28].

Hindemith observes that the traditional (tonal) harmony is largely contained in Subgroups I, II, V and VI; the vast space of Subgroups III and IV lies beyond the boundaries of tonal harmony. In his view therefore, the 'chromatic' harmony of CofMC based on the *12-tone scale* represents a considerable extension of the traditional *major/minor* harmonic theory.

In general, any move from lower to higher-numbered subgroups induces a decrease in **tonal value**, while a step in the opposite directions causes an increase. At the same time, any such increase in tonal value decreases **harmonic tension** and vice-versa. Hindemith introduces the term **harmonic fluctuation** to denote such variations in harmonic tension. Variations may be gradual or abrupt, according to the relative tonal values of successive chords, and is visualized by means of subgroup numerals and a sort of harmonic crescendo/decrescendo, as in Figure 10.

In every chordal progression, the two leading lines are the bass and the next most important one above it. Their combination is called the **two-voice framework** and determines the general

---

[27]The latter correspond to our familiar diminished triads and diminished seventh chords, while the former correspond more or less to augmented triads in tonal harmony.

[28]It is important to note that the guide-tone is not a feature of some *key* or mode (like the leading note in tonal theory) but an inherent characteristic of a *chord* containing a tritone. This again is an original invention, without any 'tonal' counterpart.

direction of harmonic activity. Intermediate lines are not relevant at this level; their purpose is to furnish each chord with the intra-chord intervals needed to determine its precise place in the chord hierarchy, in this way regulating harmonic fluctuation from one chord to the next. The **degree progression** is another vital feature, in some sense a codification of the whole sequence[29]. Indeed, a bad degree sequence usually indicates a faulty chord progression. As expected, intervals between successive chord degrees are governed by Series 2. Figure 10 demonstrates the three forces at play. The reader may observe that the degree progression in this case is made up of a single repeated pitch-class (C). The two-voice framework exhibits a peak at the fourth chord (minor seventh), followed by a release and a higher peak at the penultimate chord (tritone). The harmonic fluctuation follows a smoother line, keeping a steady medium tension for most of the progression before too stepping up to its highest point at the penultimate chord. In general, there is a constant interaction between these forces, working towards the same direction or balancing themselves out in various ways.

The degree progression is most important in setting up the 'key' of a musical phrase. The term *key* is obviously not used here as in the major/minor schema. It rather refers to the establishment of a fundamental tone, the 'key' (or **tonic center**), standing prominently in relief with the other tones in the sequence arranging themselves beside it according to their place in *Series 1*. A loss of balance in this sense may bring about the gradual or abrupt rise of a new fundamental tone and thus impose a **modulation**. In practice, the degree progression can be used along with (or sometimes against) the harmonic fluctuation in order to provide variety and purpose to the music texture. For instance, a degree progression may be blurred by superposing complex chords from Subgroups III and IV, in which case degree progression and harmonic fluctuation work along conflicting directions. Local tonic centers may form another sequence at a higher level: a fundamental tone will emerge in the same way, this time indicating the tonic center of a more extended excerpt, related to the tonic centers of its various subsections.

Hindemith has undoubtedly erected an original and, in many respects, even ground-breaking harmonic theory. Naturally, his unyielding rejection of tonal harmony does sound questionable when analyzing music obviously based on tonal precepts, despite his efforts to anticipate and nullify critics: "... to musicians irretrievably engrossed in conservatism, not only will methods of composition following our ideas appear fantastic and unartistic; it will even be denied that they are workable" [36]. On the other hand, this theory is not restricted to some particular idiom; instead, it aspires to encompass a wide range of genres from medieval to modern music, attempting to provide universal rational principles for solid music making. Particularly noteworthy is Hindemith's quantization of harmonic tension through a taxonomy of tone simultaneities, an area virtually unexplored within the tonal framework. At any rate, putting aside any music-theoretical reservations, our work focuses exclusively on the ontological aspect of his chord theory.

---

[29]Hindemith compares the usage of chord degrees to that of logarithms in mathematics.

C. Methods to extend *ontocomc* classes

The *Python/Owlready* scripts contain a number of methods to extend the functionality of *ontocomc:Interval*, *ontocomc:Chord* and *ontocomc:ChordSequence*. The most important of them are displayed below in pseudocode:

C.1. Methods extending *ontocomc:Interval*

def root(*self*):
 if *self* is classified as *RootLow* return *self.low*
 if *self* is classified as *RootHigh* return *self.high*
 else return *ontocomc.PitchClassUndef*
def series_2(*self*):
 series2_intvs = [Unison, Perf5th, ..., Tritone] # CofMC's Series-2
 find the type of *self*, then return its rank in *series2_intvs* # values in (1,…,12)
 if *not found* return 13

C.2. Methods extending ontocomc:Chord

def notes(self):
 return [self.note1, self.note2, self.note3, self.note4]
def intrachord(self):
 for i in range(len(self.notes())):
  for j in range(i+1,len(self.notes())):
   create individual *ontocomc.<self.name>_(i+1)(j+1)*
 run the reasoner to classify new intra-chord interval individuals
 for each intra-chord interval *x* calculate *x.series_2()*
 for $X, Y = 1, 2, 3, 4$ and $X < Y$:
  self.intervalXY = <self_name>_XY
 return interval names, Series-2 values (ordered by Series-2 values)
def find_degree(self,intrachord_dataframe):
 # intrachord_dataframe is the output of self.intrachord()
 # returns an integer indicating the position of the degree in the chord
 if *self* in {SubgroupV, SubgroupVI}:
  self.degree = ontocomc.PitchClassUndef
  return 0
 else:
  drop top line/s of intrachord_dataframe with *Series-2 value* = 1
  self.degree = x for *x*: root of top line interval
  return position (1,2,3,4) of chord degree
def find_guidetone(self,degree_position,intrachord_dataframe):
 # returns an integer indicating the position of the degree in the chord
 if *self* in {SubgroupV, SubgroupVI}:
  self.guideTone = ontocomc.PitchClassUndef

```
            return 0
       else:
           if self in Group A:
               self.guideTone = self.degree
               return chord degree position
           else:
               single out all tritone notes
               find tritone note x closest (by Series-2) to chord degree
               self.guideTone = x
               return position (1,2,3,4) of guide-tone
   def salient_dissonance(self,intrachord_dataframe):
       # returns a tuple with the note positions of the most prominent dissonance
       restrict intrachord_dataframe to intervals with Series-2 values in [8,12]
       if self contains no dissonant intervals: return (0,0)
       else:
           extract index (X,Y) from bottom line of (restricted) intrachord_dataframe
           self.salientDissonance = ontocomc.<self_name >_XY
           return (X,Y)
   def derove (self,neighbor):
       # neighbor may be the chord preceding or following the roving harmony
       # returns a tuple with the degree and guide-tone of roving chord
       if neighbor.degree == ontocomc.pitchClassUndef: return (0,0)
       else:
           create fresh intervals between each of self.notes() and neighbor.degree
           run the reasoner to classify new intervals, compute their Series-2 values
           select interval x with min Series-2 value
           self.degree = x.high
           return (y,0), where y: position of self.degree in the chord
```

C.3. Methods extending ontocomc:ChordSequence

```
def seq_features(self):
   # updates the ontology with chord features, i.e. degrees, guide-tones etc.
   # applies rationale of ontocomc.Chord methods for all chords in one batch
   for chord in sequence:
       create fresh individuals for intra-chord intervals
       chord.intervalXY = <chord_name >_XY
   classify new intervals # only once for all intra-chord intervals for all chords
   for chord in sequence:
       find degree, guide tone and salient dissonance # apply chord methods
   return degrees, guide-tones and salient dissonances
def seq_derove(self,analysis_results):
   for chord in sequence:
       if exists chord.degree or chord == first/last in sequence: continue
```

           elif exists *<preceding_chord>.degree*: *neighbor = <preceding_chord >*
           elif exists *<following_chord>.degree*: *neighbor = <following_chord >*
           else: continue # chord will remain at roving state
        for each roving *chord*:
           identify *chord.degree* # apply *chord.derove()*
def seq_analysis(self):
        self.seq_features() # to obtain analysis_results
        self.seq_derove(analysis_results)

D. The chord sequence generation procedure

The chord sequence generation pre-processing step (cf. Chapter 5) is displayed below in pseudocode. The procedure receives at the input a Bach chorale excerpt in MIDI music file format and generates 4 sequences of pitch-classes, representing the four voice lines, suitably unfolded at quaver quantization:

    transcribe input MIDI file to CSV format # using *py_midicsv* Python package
    store info in Pandas DataFrame *data_frm*
    for *channel* in channels 2,3,4,5: # soprano, contralto, tenor and bass respectively
        restrict *data_frm* to entries with "channel"==*channel*
        drop first, second and last lines of *data_frm*
        drop column "channel"
        assign *quaver_nr*. Keep entries with *timestamp = 512*n*
        extract pitch-classes from MIDI pitch numbers
        tied = False # default value
        for *quaver_nr* between successive *note_on* and *note_off* entries:
           pad with pitch-class issued at *note_on* timestamp
           tied = True
        for *quaver_nr* indicating a musical pause between successive *note_off* and *note_on* entries:
           pad with *ontocomc.pitchClassUndef*
        set *quaver_no* as DataFrame.Index
    # the 4 voices should now be of equal length
    for *quaver_no* in DataFrame.Index:
        instantiate *chord* with respective 4 pitch-classes, octaves and *tied* information
        if *quaver_no* == odd:
           chord ontocomc.beat 'on' # chord is on accented part of bar
        else:
           chord ontocomc.beat 'off'
    instantiate *chord_seq*
    for *chord*, *chord_next* in *chord_seq*: # connect chords as members of sequence
        chord ontocomc.nextChord chord_next
        chord_last ontocomc.nextChord ontocomc.chordNil