MUSIC AS FLOW: A FORMAL REPRESENTATION OF HIERARCHICAL PROCESSES IN MUSIC

Zeng Ren EPFL zeng.ren@epfl.ch
Wulfram Gerstner EPFL wulfram.gerstner@epfl.ch
Martin Rohrmeier EPFL martin.rohrmeier@epfl.ch

ABSTRACT

Modeling the temporal unfolding of musical events and its interpretation in terms of hierarchical relations is a common theme in music theory, cognition, and composition. To faithfully encode such relations, we need an elegant way to represent both the semantics of prolongation, where a single event is elaborated into multiple events, and process, where the connection from one event to another is elaborated into multiple connections. In existing works, trees are used to capture the former and graphs for the latter. Each such model has the potential to either encode relations between events (e.g., an event being a repetition of another), or relations between processes (e.g., two consecutive steps making up a larger skip), but not both together explicitly. To model meaningful relations between musical events and processes and combine the semantic expressiveness of trees and graphs, we propose a structured representation using algebraic datatype (ADT) with dependent type. We demonstrate its applications towards encoding functional interpretations of harmonic progressions, and large scale organizations of key regions. This paper offers two contributions. First, we provide a novel unifying hierarchical framework for musical processes and events. Second, we provide a structured data type encoding such interpretations, which could facilitate computational approaches in music theory and generation.

1. INTRODUCTION

When understanding music as a temporal art, there are at least two properties we need to model. The first is that musical events are ordered in a nontrivial way resembling goal-directedness; the essence of a piece is lost if we "recompose" a piece by performing random temporal permutations. The second phenomenon is the temporal hierarchy, which is a central theme in the understanding of Western tonal music, where we hear multiple entities as the manifestation of a single musical entity. Regarding this hierarchy, there are at least two kinds of such entities. The first kind is a stationary process, such as key region, and harmony; we can say a phrase that enforces a key contains a tonic region (like some presentation in a non-modulating sentence) and we can also describe a time span as an arpeggiation of a harmony. The second kind is a transitory process, such as modulation region, passing, and neighboring motion; a descending third progression contains two stepwise downward motions.

There are multiple attempts to represent the hierarchical structure of such entities. For stationary entities, trees of musical events have been used to model tonal harmony [1, 2], extended tonal harmony [3], jazz harmony [4], rhythm and meter [2, 5]. One limitation of using trees of musical events is that semantics such as passing tones could not be elegantly expressed because one is forced to select either the left or the right parent event for the subordinate event whereas we would like to express an intermediate event subordinate to the melodic motion itself [2].

For the transitory process, trees on event transitions are also sometimes used [6]. They could model the semantics for a passing tone by describing how a melodic motion is split into two motions, one going to the passing note and one leaving the passing note. However, as the fundamental entities are transitions, it can not express the idea that a single event being elaborated in the temporal dimension, such as unfoldings, complete neighbor chords/tones, repetitions, and rearticulations [7].

There are attempts using graphical notations to capture both stationary and transitory processes [7–9]. There are also models [10] that extend such hierarchical organizations beyond the temporal dimension with inner structures of events resembling concurrent processes.

One could potentially encode the hierarchical organization between these two kinds of processes implicitly using networks and graphs, or more expressively using hypergraphs where higher-order relations can be encoded as hyper-edge. One could formulate a rewrite grammar on such networks and hypergraphs to describe the elaboration of nodes and edges. However, we believe there should be a more direct, elegant, and specialized solution (in a similar spirit as [11]) to not only implement but also characterize such generative principles of hierarchical processes.

In summary, there is a lack of formal representation as well as a specially designed data structure that explicitly captures the intricate hierarchical organizations of both the stationary and transitory processes, a fundamental idea in reductive theories of tonal music. This paper offers two contributions. First, we provide a novel unifying hierar-

chical framework for stationary and transitory processes. Second, we provide a structured data type encoding such interpretations, which could facilitate computational approaches in music theory, musicology, and algorithmic music composition.

2. THE HIERARCHICAL ORGANIZATIONS OF GENERAL PROCESSES

To demonstrate how these two kinds of processes could be hierarchically organized in the temporal dimension, perhaps it is helpful to consider a scenario in everyday life: "On his way back to home, John went to the supermarket, where he got his favorite yogurt from the fridge. Although he could take a bus directly to his house, he decided to get off one stop earlier by the lake to enjoy a short walk." One hierarchical organization of this particular scenario is depicted in Fig.1. The overarching process is that John went back home from someplace. This transitory process (represented by the arrow connecting "someplace" denoted by X to "home") contains three component processes: a transitory process from "someplace" to "supermarket", a stationary process at "supermarket," and a transitory process from "supermarket" to "home." The stationary process at "supermarket" further contains a stationary process at the "entrance" of the supermarket, a transitory process from "entrance" to the "exit," and a stationary process at the "exit".

2.1 The syntactic constraint of the hierarchical organization of stationary and transitory processes

One pattern that we observe is the mutual recursive relationship between stationary and transitory processes. A stationary process can contain three components (stationary, transitory, stationary). Symmetrically, a transitory process can contain three components (transitory, stationary, transitory).

However, it is clear from the above example (Fig. 1) we can not arbitrarily subdivide a stationary process at X (denoted by $\widehat{X}$), or a transitory process from X to Y (denoted by $\xrightarrow{X \to Y}$) into arbitrary triples of processes, even if they conform to the (stationary, transitory, stationary) or (transitory, stationary, transitory) patterns. We may allow a transitory process $\xrightarrow{A \to B}$ to be elaborated into three components of the form

$$\xrightarrow{A \to X} \widehat{X} \xrightarrow{X \to B}$$

But we would not allow a decomposition like

$$\xrightarrow{C \to D} \widehat{X} \xrightarrow{E \to F}$$

because their states are not compatible.

We can summarize the constraints as the following: a stationary process $\widehat{X}$ may contain ($\widehat{X}$, $\xrightarrow{X \to X}$, $\widehat{X}$); likewise, a transitory process $\xrightarrow{X \to Y}$ may contain ($\xrightarrow{X \to Z}$, $\hat{Z}$, $\xrightarrow{Z \to Y}$). The "entrance" and "exit" in the previous example, although being technically different, are equivalent to "supermarket" from abstract level.

3. LINEAR PROCESSES

We start with characterizing linear processes representing a single hierarchical stream.

3.1 An axiomatic system

We refer to a stationary process as $Joint$, and define it as a predicate J_x indexed by a state $x : A$. We refer to a transitory process as $Link$ and define it as a predicate $L_{x,y}$ indexed by two states x, y of the same type. Then we propose the following four axioms to characterize the hierarchical interactions between of stationary and transitory processes.

$$\forall(x : A)\ \exists(j : J_x) \tag{1}$$
$$\forall(x, y : A)\ \exists(l : L_{x,y}) \tag{2}$$
$$\forall(j, j' : J_x)\ \forall(l : L_{x,x})\ \exists(j^* : J_x) \tag{3}$$
$$\forall(l : L_{x,z})\ \forall(j : J_z)\ \forall(l' : L_{z,y})\ \exists(l^* : L_{x,y}) \tag{4}$$

Axiom 1 states that we may form a stationary process for a given state. Axiom 2 states that we may form a transitory process by for a pair of states of the same kind. Axiom 3 states that we may form a stationary process at x from any triple of processes (j, l, j'), where j and j' are both of stationary processes at x and l is a loop starting and ending at x . Axiom 4 states that we may form a transitory process from any triple of processes (l, j, l'), where l and l' are transitory and j is stationary, provided that their states are compatible.

3.2 A syntax based on dependent type theory

Using two mutually inductive algebraic datatypes, $Joint$ and $Link$, we formalize the notion of hierarchical process in Backus–Naur form (Eq. 5,6,7,8). For a stationary process, the base case (Eq. 5) of a $Joint$ is a $Point$, which means an atomic stationary process, whereas the inductive case (Eq. 6) resembles a stationary process (on the current level) containing two stationary process and a transitory process. The base case of a $Link$ is a $Unit$ (Eq. 7), representing a indivisible change of state, whereas the inductive case (Eq. 8) represents a composite motion that contains two changes and one stationary process. Eq. (5,6,7,8) corresponnds to Axiom. (1,3,2,4) respectively.

$$Joint_{(x:a)} := \langle Point \rangle \quad (x : a) \tag{5}$$
$$\mid \langle Joint \rangle \quad Joint_x \quad Link_{x,x} \quad Joint_x \tag{6}$$

$$Link_{(x:a),(y:a)} := \langle Unit \rangle \quad (x : a) \quad (y : a) \tag{7}$$
$$\mid \langle Link \rangle \quad Link_{x,z} \quad Joint_z \quad Link_{z,y} \tag{8}$$

Dependent typing [12] allows us to define types depending on value. This algebraic data structure with dependent typing has an important application for a generative system. One can define a function using polymorphic recursion to sample a value of the given type. Do-

$$\hat{X} \xrightarrow{X\to h} \widehat{home}$$
$$\hat{X} \xrightarrow{X\to s} \widehat{supermarket} \xrightarrow{s\to h} \widehat{home}$$
$$\hat{X} \xrightarrow{X\to s\sim en} \widehat{entrance} \xrightarrow{en\to ex} \widehat{exit} \xrightarrow{ex\sim s\to l} \widehat{lake} \xrightarrow{l\to h} \widehat{home}$$
$$\hat{X} \xrightarrow{X\to en} \widehat{entrance} \xrightarrow{en\to f} \widehat{fridge} \xrightarrow{f\to ex} \widehat{exit} \xrightarrow{ex\to l} \widehat{lake} \xrightarrow{l\to h} \widehat{home}$$

Figure 1: A hierarchical interpretation of John's journey. Words on the transitions are abbreviated to save space. The symbol $s \sim en$ means the "entrance" is functionally equivalent to "supermarket" in the interpretation of this journey

.

ing so will guarantee the syntactic correctness of the output. For example, writing a harmonic transition between I and V means sampling a value of type $Link_{I,V}$; elaborating a melodic motion from $\hat{8}$ to $\hat{5}$ becomes sampling a value of type $Link_{\hat{8},\hat{5}}$. Note that between these two examples, the type of the state is different; the first is harmony(roman-numeral) while the second is scale degree. However, within each example, the types of the states are always the same by construction (Axiom 2, 3, 4).

3.3 A data structure in Haskell

Haskell is a functional programming language with an expressive type system [13]. Although the dependent type is not yet built into the language, there exists an encoding involving Algebraic Datatype and Singletons [14] to simulate the behavior of dependent type. The implementation of the linear process is shown below [1] .

```
data Joint (x::a)
    = Point (Sing x)
    | Joint (Joint x) (Link x x) (Joint x)

data Link (x::a) (y::a)
    = Unit (Sing x) (Sing y)
    | forall (z::a). Link (Link x z) (Joint z) (Link z y)
```

3.4 A graphical notation system

To visualize an interpretation of hierarchical processes, we use two types of slurs to connect states in a sequence. For a $Point$, the base case of $Joint$, no visual representation is needed as the state it expresses is sufficient. For a $Unit$, the base case of $Link$, a dashed slur is drawn connecting the starting state to the ending state. Nontrivial stationary processes (the inductive cases) are represented as continuous slurs whereas transitory processes are represented with dashed slur. For a non-trivial processes, the left/right anchor point of the slur is the same as the left/right anchor point of the slur of the process's first/last component.

Five simple examples of such notation are provided in Fig. 2. The easiest to understand is Ex. D, which is a stationary process $\hat{I}$, decomposed into a $\hat{I} \xrightarrow{I\to I} \hat{I}$. This expresses not just repetition but also the loop from I to itself. This loop can be the parent for further elaborations such as passing and neighbor chords. Note that the continuous slur covers a point, a dashed slur, and a point. This visual "covering" is intend to convey the hierarchical relation of processes where parent processes contains its component processes. Ex. C shows a more complex situation; although the top two level is the same as Ex. D (a $Joint$ containing two base case $Joint$ and a $Link$), there are richer structures within the $Link$ in Ex. C. This dashed slur covers a dashed slur, a continuous slur, and a dashed slur, signaling a non-trivial transitory process that contains a transitory process $\xrightarrow{I \quad V}$, a stationary process at V, and a transitory process $\xrightarrow{V \quad I}$. This stationary process at V is also a non-trivial one, capturing the sense of prolongation within which passing notes can be generated connecting its chordal pitches, forming a harmonic the entity $\mathrm{V}^{7-6-5}_{5-4-3}$. Ex. B expresses an overarching stationary process at I containing a nontrivial initial stationary process, a nontrivial transitory process and a trivial stationary process. The initial stationary process resembles a neighbor-passing chord with in the harmony of I. Note that there is a difference in semantic in drawing the continuous slur over the first two Is vs the last two Is. The former means that the overarching motion is stationary first and then moving to the target I chord, whereas the latter means the motion moves first and then performs a stationary rest. This kind of fine distinction could be used to further express embodied musical concepts such as "momentum," "potential energy," and "forces" [15]. Ex. A presents two interpretations of the same chord progression, where the top and bottom analysis reflects the melody and the bass respectively.

4. APPLICATIONS IN MUSIC ANALYSIS

We will demonstrate the potential usage of the proposed model in reductive analysis in three different levels of complexity.

4.1 The harmonic sequence of a hybrid theme type

For a harmonic sequence $I - V - V - I - IV - V - I$ in an 8-bar phrase, we may interpret the first four chords as a tonic prolongation and the rest as an incomplete cadential progression, as in an antecedent + cadential hybrid theme type [16]. Such an interpretation is captured by the derivation process in Fig. 3.

[1] Specifying the singleton arguments can be sometimes redundant and cumbersome. Instead, one could use the "implicit-passed" singleton by replacing the argument "Sing x" by a constraint "SingI x". In this way, the "Point" and "Unit" constructor takes no explicit argument and the state information is thus encoded as a phantom type that can be pattern matched using type application such as "Point @IV".

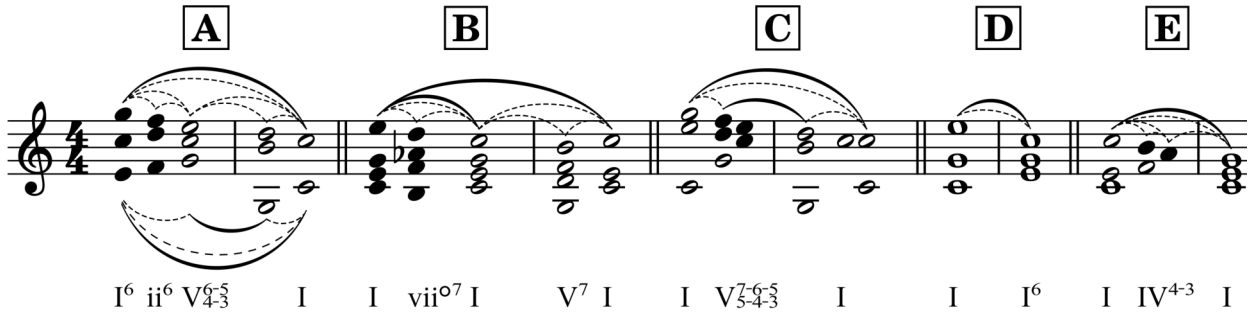

Figure 2: Examples of the graphic notation for Hierarchical processes where continuous slurs represent $Joint$ and dashed slurs represent $Link$.

$$\widehat{I}$$

$$\widehat{I} \xrightarrow{I \to I} \widehat{I}$$

$$\widehat{I} \xrightarrow{I \to I} \widehat{I} \xrightarrow{I \to V} \widehat{V} \xrightarrow{V \to I} \widehat{I}$$

$$\widehat{I} \xrightarrow{I \to V} \widehat{V} \xrightarrow{V \to I} \widehat{I} \xrightarrow{I \to IV} \widehat{IV} \xrightarrow{IV \to V} \widehat{V} \xrightarrow{V \to I} \widehat{I}$$

Figure 3: One derivation process for the harmonic sequence $I - V - V - I - IV - V - I$. The first three steps are elaboration of types whereas the last step is the instantiations of the types (in the framework of formal grammar, this corresponds to rules generating nonterminal and terminal symbols)

4.2 The key-level modulation analysis of a simple ternary form

Now we present an analysis of a section of Haydn, Piano Sonata in D, H.37, iii, using the proposed model (Fig. 4). The main focus of the analysis here is on key center and functional harmony. This overall section resembles a stationary process at the home key region. It contains a stationary process expressing the home key region, followed by a transitory process from to a home key region to another home key region. Within this transitory process, there is a stationary process at the dominant key region, as well as the two transitory processes functioning as key transition. The first connects the tonic chord in the home key $I\{I\}$ to the tonic chord in the dominant key $V\{I^6\}$ as the pivot chord for modulation. The second connects $I\{V\}$ to $I\{I\}$ as to signal the return to the home key. In a prolongational framework of pitch reduction, this link at mm.12 is the interrupted motion in a typical interruption, implying the restart of the fundamental line [8]. In a functional harmony framework, this same link is the dominant to tonic preparation relation, also implying the arrival of the tonic in the home key. Within the dominant region the first stationary process corresponds to the complete cadential progression signaling the stabilization of the dominant key, whereas the following transitory process $\xrightarrow[V\{I\} \to ||I\{V\}]{\hat{5} \to ||\hat{1}}$ corresponds to first attempted descent (interrupted) of the fundamental line accompanied by the "Ponte" modulation schema [17] that gradually change the underlying key of a chord.

4.3 A harmonic analysis of a Bach chorale

A more elaborated example of such harmonic organization can be found in Bach Chorale No 9 BWV 248 (Fig. 5). This analysis interprets the overarching structure of the piece as a stationary process in the home key tonic, containing an establishment of the home key, detour around the home key, and the re-stablization of the home key. This transitory process $\xrightarrow[I\{I\} \to I\{I\}]{\hat{1} \to \hat{8}}$ enforces its unity with an ascending linear progression of an octave, within which the music modulates to the relative minor (vi) via its dominant minor key (V/vi). The first stationary process around the home key is elaborated into a $I - V - I$ ternary-like structure on the key level.

5. CONCURRENT PROCESSES

Besides providing a formalization for processes in the temporal dimension, we also offer an extension to model the processes that are simultaneous, which enables us to model polyphonic texture.

5.1 The Formalism

To model concurrent processes, we add two **non-commutative** binary [2] operations to construct $Joint$ and $Link$ respectively:

$$(\infty) : Joint_x \to Joint_y \to Joint_{(x,y)} \quad (9)$$

$$(\Uparrow) : Link_{x,y} \to Link_{x',y'} \to Link_{(x,x'),(y,y')} \quad (10)$$

The first constructor ∞, called "**when**", expresses the concurrency of two stationary processes. The second constructor $\Uparrow$, called "**while**", expresses the concurrency of two transitory processes. Applying these operators using infix

[2] Although these operations are currently formulated as binary operations, they can be naturally extended to n-nary versions where the input is a vector of $Joint$ and $Link$ respectively.

notations, we have Eq. 11, 12:

$$\widehat{x} \infty \widehat{y} := \widehat{(x, y)} \tag{11}$$

$$\xrightarrow{x \to y} \Uparrow \xrightarrow{x' \to y'} := \xrightarrow{(x,x') \to (y,y')} \tag{12}$$

In addition we add an algebraic law (Eq. 13) on these two operations:

$$\begin{gathered}(\widehat{x} \infty \widehat{y})(\xrightarrow{x \to x} \Uparrow \xrightarrow{y \to y})(\widehat{x} \infty \widehat{y}) \\ = \\ (\widehat{x} \xrightarrow{x \to x} \widehat{x}) \infty (\widehat{y} \xrightarrow{y \to y} \widehat{y})\end{gathered} \tag{13}$$

To model temporal displacement of concurrent processes (like the suspension in fourth species counterpoint) we define a function $\uparrow_{\uparrow}$ called "**leads**" that is derivable from the basic operations in terms of Eq. 9,10 :

$$(\uparrow_{\uparrow}) : Link_{x,y} \to Link_{x',y'} \to Link_{(x,x'),(y,y')}$$

$$\xrightarrow{x \to y} \uparrow_{\uparrow} \xrightarrow{x' \to y'} = \xrightarrow{(x,x') \to (y,x')} \widehat{(y, x')} \xrightarrow{(y,x') \to (y,y')} \tag{14}$$

$$= (\xrightarrow{x \to y} \Uparrow \xrightarrow{x' \to x'})(\widehat{y} \infty \widehat{x'})(\xrightarrow{y \to y} \Uparrow \xrightarrow{x' \to y'}) \tag{15}$$

With Eq. 14, we formalize the relation about two processes where that one process **leads** the other. For example, in a typical suspension, this allows us to capture the meaning that the bass motion leads the melody motion, causing a consonance-dissonance-consonance pattern.

5.2 Concurrent processes in contrapuntal textures

Now we demonstrate that using the proposed formalism, we can model many complex hierarchical coordination of processes in both temporal (horizontal) and spatial (vertical) dimensions.

5.2.1 First species counterpoint

For first species counterpoint, processes are always vertically aligned in a one-to-one fashion.

$$\begin{array}{ccccc} 3 & - & 2 & - & 1 \\ 1 & - & 7 & - & 1 \end{array}$$

can be modeled as:

$$\widehat{(3,1)}\Big(\big(\xrightarrow{3 \to 2} \Uparrow \xrightarrow{1 \to 7}\big)\widehat{(2,7)}\big(\xrightarrow{2 \to 1} \Uparrow \xrightarrow{7 \to 1}\big)\Big)\widehat{(1,1)}$$

5.2.2 Second species counterpoint

For second species counterpoint, we encounter concurrent processes where one is more elaborated than the other. A segment of such texture

$$\begin{array}{ccccc} 5 & - & 4 & - & 3 \\ 7 & & - & & 1 \end{array}$$

can be modeled as

$$\widehat{(5,7)}\big((\xrightarrow{5 \to 4} \widehat{4} \xrightarrow{4 \to 3}) \Uparrow (\xrightarrow{7 \to 1})\big)\widehat{(3,1)}$$

Notice that the two-against-one coordination of the motion is reflected in the structure of the encoding and we do not need to "break" the $\hat{7}$ into two copies of $\hat{7}$ to convert it into first species texture.

5.2.3 Third species counterpoint

Third species counterpoint is a more elaborated version of the second species but the form of the representation is very similar.

5.2.4 Fourth species counterpoint

Fourth species counterpoint presents the opportunity of suspension. We can generalize such textures as overlaying transitory processes in an alternating manner, creating temporal displacement. Consider this 7-6 suspension ("=" represents "tied-over")

$$\begin{array}{c} 3 = 3 - 2 = 2 - 1 \\ 5 - 4 = 4 - 3 = 3 \end{array}$$

It can be modeled using Eq. 14 as the following:

$$\widehat{(3,5)}\Big(\big(\xrightarrow{5 \to 4} \uparrow_{\uparrow} \xrightarrow{3 \to 2}\big)\widehat{(2,4)}\big(\xrightarrow{4 \to 3} \uparrow_{\uparrow} \xrightarrow{2 \to 1}\big)\Big)\widehat{(1,3)}$$

6. DISCUSSION

The contribution of this paper is to offer a characterization and representation on the hierarchical organization of both stationary and transitory musical processes as well as how they can be concurrently structured. Linear processes are modeled using two mutually inductive types $Joint$ and $Link$. Concurrent processes are modeled on top of linear processes by adding two binary operations for $Joint$ and $Link$ respectively. In addition, an algebraic law is imposed on these two operators to express an isomorphism between the horizontal view and the vertical view. We introduced a graphical notation for linear processes and presented several harmonic analysis using the notation to demonstrate the music analytical application of the characterization of linear processes. To demonstrate the analytical application of the concurrent processes, we presented their corresponding encoding for contrapuntal textures in species counterpoint.

This general formalism can be flexible to adapt to specific music theoretical constraints. One might encode specialized elaboration rules by equipping the $Link$ constructor with domain specific constraints on the type level and reuse the function itself. In Eq. 8, such constraint could be a predicate on the type variables $p_{x,z,y}$ [3] .

[3] Upper neighbor elaboration, for example, corresponds to the predicate $p_{x,z,y} = (x = y, z =\uparrow x)$. Likewise, downward passing elaboration corresponds to the predicate $p_{x,z,y} = (z =\downarrow x, y =\downarrow z)$.

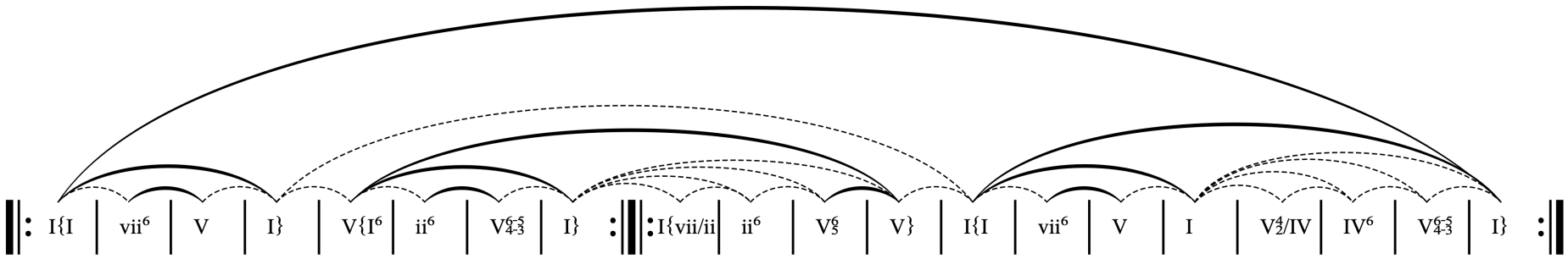

Figure 4: A prolongational analysis of Haydn, Piano Sonata in D, H.37, iii, mm. 1-20 using the notation of hierarchical process. Key regions are indicated by roman numerals followed by curly brackets.

Figure 5: An analysis of Bach Chorale No 9 BWV 248.

7. ACKNOWLEDGEMENT

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under grant agreement No 760081 – PMSB. We thank Claude Latour for supporting this research through the Latour Chair in Digital Musicology.

We thank the team of the Digital and Cognitive Musicology Lab (DCML), particularly Gabriele Cecchetti and Xinyi Guan, for their helpful comments on earlier versions of this paper. Furthermore, we thank Yannis Rammos and Christoph Finkensiep for fruitful discussions on the music interpretations and implementation of the model in this paper.

8. REFERENCES

[1] M. Rohrmeier, "Towards a generative syntax of tonal harmony," *Journal of Mathematics and Music*, vol. 5, no. 1, pp. 35–53, 2011.

[2] F. Lerdahl and R. S. Jackendoff, *A Generative Theory of Tonal Music, reissue, with a new preface*. MIT press, 1996.

[3] M. Rohrmeier and F. C. Moss, "A formal model of extended tonal harmony," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, no. CONF, 2021, pp. 569–578.

[4] M. Rohrmeier, "The syntax of jazz harmony: Diatonic tonality, phrase structure, and form," *Music Theory and Analysis (MTA)*, vol. 7, no. 1, pp. 1–63, 2020.

[5] ——, "Towards a formalization of musical rhythm." in *ISMIR*, 2020, pp. 621–629.

[6] É. Gilbert and D. Conklin, "A probabilistic context-free grammar for melodic reduction," in *Proceedings of the International Workshop on Artificial Intelligence and Music, 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 83–94.

[7] P. Westergaard, *An introduction to tonal theory*. Norton New York, 1975.

[8] H. Schenker, *Free Composition: Volume III of new musical theories and fantasies*. Pendragon Press, 2001, vol. 1.

[9] J. Yust, *Organized time: rhythm, tonality, and form*. Oxford University Press, 2018.

[10] C. Finkensiep, "The structure of free polyphony," p. 321, 2023. [Online]. Available: http://infoscience.epfl.ch/record/300206

[11] A. Mokhov, "Algebraic graphs with class (functional pearl)," *SIGPLAN Not.*, vol. 52, no. 10, p. 2–13, sep 2017. [Online]. Available: https://doi.org/10.1145/3156695.3122956

[12] M. Hofmann and M. Hofmann, "Syntax and semantics of dependent types," *Extensional Constructs in Intensional Type Theory*, pp. 13–54, 1997.

[13] S. Marlow *et al.*, "Haskell 2010 language report," *Available online http://www. haskell. org/(May 2011)*, 2010.

[14] R. A. Eisenberg and S. Weirich, "Dependently typed programming with singletons," *ACM SIGPLAN Notices*, vol. 47, no. 12, pp. 117–130, 2012.

[15] S. Larson, *Musical forces: Motion, metaphor, and meaning in music*. Indiana University Press, 2012.

[16] W. E. Caplin, *Analyzing classical form: An approach for the classroom*. Oxford University Press, USA, 2013.

[17] R. Gjerdingen, *Music in the Galant Style*. OUP USA, 2007.