"Towards a Practical Tool for Music Composition: Using Constraint Programming to Model Chord Progressions and Modulations"

Sprockeels, Damien ; Van Roy, Peter

ABSTRACT

The Harmoniser project aims to provide a practical tool to aid music composers in creating complete musical works. In this paper, we present a formal model of its second layer, tonal chord progressions and modulations to neighbouring tonalities, and a practical implementation using the Gecode constraint solver. Since music composition is too complex to formalize in its entirety, the Harmoniser project makes two assumptions for tractability: first, it focuses on tonal music (the basis of Western classical and popular music); second, it defines a simplified four-layer composition process that is relevant for a significant number of composers. Previous work on using constraint programming for music composition was limited to exploring the formalisation of different musical aspects and did not address the overall problem of building a practical composer tool. Harmoniser's four layers are global structure (tonal development of the whole piece), chord progressions (diatonic and chromatic) and modulations, voicing (four-voice chord layout), and ornaments (e.g., passing notes, appoggiaturas), all allowing iterative refinement by the composer. This paper builds on prior work for voicing layer 3, \textit{Diatony}, and presents a model for layer 2, chord progressions and modulations. The results of the present paper can be used as input to \textit{Diatony} to generate voicing. Future work will define models for the remaining layers, and combine all layers together with a graphical user interface as a plug-in for a DAW.

CITE THIS VERSION

Towards a Practical Tool for Music Composition: Using Constraint Programming to Model Chord Progressions and Modulations

Damien Sprockeels, **Peter Van Roy**
ICTEAM, UCLouvain, Louvain-la-Neuve, Belgium
{damien.sprockeels, peter.vanroy}@uclouvain.be

Abstract

The Harmoniser project aims to provide a practical tool to aid music composers in creating complete musical works. In this paper, we present a formal model of its second layer, tonal chord progressions and modulations to neighbouring tonalities, and a practical implementation using the Gecode constraint solver. Since music composition is too complex to formalize in its entirety, the Harmoniser project makes two assumptions for tractability: first, it focuses on tonal music (the basis of Western classical and popular music); second, it defines a simplified four-layer composition process that is relevant for a significant number of composers. Previous work on using constraint programming for music composition was limited to exploring the formalisation of different musical aspects and did not address the overall problem of building a practical composer tool. Harmoniser's four layers are global structure (tonal development of the whole piece), chord progressions (diatonic and chromatic) and modulations, voicing (four-voice chord layout), and ornaments (e.g., passing notes, appoggiaturas), all allowing iterative refinement by the composer. This paper builds on prior work for voicing layer 3, *Diatony*, and presents a model for layer 2, chord progressions and modulations. The results of the present paper can be used as input to *Diatony* to generate voicing. Future work will define models for the remaining layers, and combine all layers together with a graphical user interface as a plug-in for a DAW.

1 Introduction

Constraint Programming (CP) is a popular technique for generation [Pachet and Roy, 2011; Papadopoulos *et al.*, 2015; Bonlarron and Régin, 2024]. Music generation with CP is popular as well, in particular for harmonisation. It is sometimes added to some form of learning [Lattner *et al.*, 2018; Giuliani *et al.*, 2023] to provide more user control as well as a better global structure. However, learning is limited by the training data. An alternative approach is to formalise music theory and rely solely on CP [Huang and Chew, 2005; Anders, 2008; Anders and Miranda, 2009; Carpentier *et al.*, 2010; Davismoon and Eccles, 2010]. This alternative has multiple advantages. First, there is no limitation based on training data. Second, it ensures that the rules are satisfied in every generated solution. Third, it makes the solutions easier to tweak, giving more control to composers. However, there are two disadvantages. First, it requires a substantial set of rules to make the generated solutions usable, and second, finding solutions often requires significant computation at runtime. Previous work using CP to create composer tools falls into two categories: Tools formalising one specific aspect of music theory [Ebcioğlu, 1990; Truchet *et al.*, 2003; Herremans and Sörensen, 2013], that lack generality, and tools that can model large amounts of music theory [Anders *et al.*, 2005; Laurson and Kuuskankare, 2005; Sandred, 2010] but require programming to be used, therefore limiting their usability for composers. In contrast, we propose an approach that completely models a musical style without requiring programming skills to be usable. Aside from CP and ML, two other approaches are generative grammars [Rohrmeier, 2011] and conceptual blending [Eppe *et al.*, 2015]. Chord-blending generates new progressions from existing ones, and it could potentially be combined with our approach to obtain more complex modulations. Generative grammars are used to understand the recursive structure of tonal harmony. Compared to these two approaches, constraint programming allows composers to add arbitrary musical ideas to a musical theory such as tonal harmony, and find coherent solutions (see Section 4.2 for a concrete example).

1.1 Harmoniser project

The Harmoniser project aims to build a practical tool to aid composers based on CP that addresses both the issues of rule definition and computation time. The rules are inferred from treatises on music theory (see Section 3). Because the solver enforces rules, this relieves the composer of much tedious work so they can focus on adding musical ideas to shape solutions into a desired result. To reduce computational complexity, we follow the decomposition of the composition process that was introduced in [Sprockeels and Van Roy, 2024], following [Pachet and Roy, 2001] which concludes that proper structuring is necessary to make constrained musical composition feasible. It decomposes the process of musical composition in four layers, resulting in smaller problems that

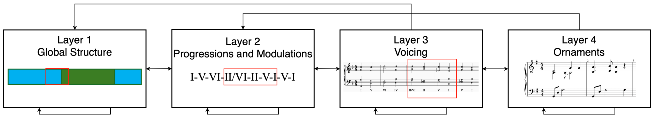


Figure 1: Illustration of the four-layer framework for composing a musical piece.

can be solved independently (see Section 2). The first layer is global harmonic structure, the second is chord progressions and modulations, the third is chord voicing, and the fourth adds ornaments. The present paper focuses on the second layer. The third layer was published in [Sprockeels and Van Roy, 2024] and defines the *Diatony* model.

1.2 Contributions

The present paper has two main contributions. First, a formal model of the second layer of the composition process, namely chord progressions in a tonality and modulations to neighbouring tonalities. This model gives freedom to express musical ideas while guaranteeing that tonal music rules are respected. Second, a constraint implementation in the Gecode constraint solver [Gecode Team, 2019]. The generated output can be given to *Diatony* to generate a voicing for the solutions. The solution space contains all possible tonal chord progressions with the given parameters, and composers express musical ideas by giving desired tonalities and modulations as well as adding constraints. This prunes the search space and eventually gives a desired musical solution.

1.3 Structure of the paper

Section 2 gives the four-layer framework for music composition, which refines the framework of [Sprockeels and Van Roy, 2024]. Section 3 defines the formal model of tonal progressions and modulations. Section 4 evaluates the model showing its intended use by a composer. Section 5 concludes this paper and outlines future work.

2 Harmoniser Project

There are probably as many ways to compose music as there are composers. In a practical tool, it is nonetheless important to give a predefined structure to guide composers. In [Sprockeels and Van Roy, 2024], an iterative process with four layers that are general enough to be relevant for a significant portion of composers was identified, and is refined in this paper. Figure 1 illustrates this structure. This paper focuses on the second layer, namely chord progressions and modulations. To put this in context, we explain all the layers.

Global structure The first layer decomposes the piece into progressions, each in a single tonality, connected by modulations. It can happen that there is only one tonality for the whole piece, and hence no modulations, but in the general case, there are multiple tonalities with one modulation between every two successive progressions. The same tonality can be present more than once in the whole piece.

Progressions and modulations The second layer realises the harmonic development within each progression. It defines a sequence of chord degrees for each of the progressions, as well as modulations to transition from one progression to the next. Chord degrees can be diatonic (belonging to the tonality) or chromatic (not belonging to the tonality). Section 3 gives the main aspects of the formal model for this layer.

Voicing The third layer defines the voicing, i.e., the actual notes on the musical staff for each chord. The typical way to represent chord voicing is to use four voices. There are two aspects to take into account: "vertical" harmony, i.e. the interaction between notes in a given chord, and "horizontal" harmony, the interaction between notes in a voice over time. Some voicing rules have an influence on chord states, in which case they are also handled in layer 2 and thus in the model of this paper. They are presented in Section 3. Layer 3 is presented in *Diatony* [Sprockeels and Van Roy, 2024].

Ornaments The fourth and final layer is melodic ornaments. Given a voicing for a chord progression, ornaments such as passing notes or appoggiaturas are added as details to enrich the musical piece. This adds essential complexity to the harmony.

3 Formal Model of Tonal Chord Progressions

We now define the formal model of tonal chord progressions and modulations, which is the second layer of the Harmoniser project and the main focus of this paper. Here, a *progression* is a sequence of chord degrees in a given tonality and a *modulation* is a transition between two successive progressions of different tonalities. The model defines the possible chords (degree, quality and state) and transitions between them following the theory of Western tonal music, which is based on the concept of *tonality*[1]. All the concepts used in the model are standard concepts of tonal music theory, for which many references exist[2].

The rules implemented in the model are taken from [Duha, 2016] and [Gauldin, 2004], ensuring consistency with *Diatony* [Sprockeels and Van Roy, 2024] that uses the same references. We use Duha's chapter on modulations as well as Gauldin's chapters 4 (triads and seventh chords), 6 (partwriting), 8-11 (diatonic harmony), 13 (dominant chords), 14 (predominant chords), 16 (6-4 chord), 17 (third and sixth degree), 19 (leading tone seventh chord), 21 (secondary dominant chords), 29 (Neapolitan chord), and 30 (augmented sixth chord). This model was established in collaboration with two composers to ensure its correctness and utility for composers.

3.1 Basic concepts of the model

The formalisation builds on the concepts of chord and chord transition:

- A *chord* is a set of three or more notes, uniquely identified by a triple (r, q, s) where r is the root note (one of the twelve notes of Western music), q is the quality (which defines the intervals between the chord notes), and s is the state (defined by the chord note that is at the

[1] A tonality is defined as a pair of a key (one of the twelve notes C, C#, D, up to B) and a mode (major or minor).

[2] Some links: tonality, chords and functions, secondary dominants and augmented sixth chords, amongst many others.

lowest voice). Within a given tonality, each note has a degree d that defines its function within the tonality as well as the possible qualities and states that are available to build chords on that note as a root.

- A *chord transition* is a pair of two chords. In Western tonal music, chord transitions are defined by degree transitions in a tonality.

Our formal model is built on top of these two concepts. The constant transition matrix T defines possible chord transitions between two chord degrees in a tonality, while three other constant matrices M, P, and L define the relationships between a chord degree and the possible chord qualities, states, and root notes respectively. These matrices compactly encode a large amount of tonal music theory, which to our knowledge has not been done by any previous composer tool. Aside from these matrices, constraints are enforced to model more specific aspects of tonal music theory that are not captured by the matrices. Additional constraints are also enforced to allow for modulations between tonalities, which is a key aspect of Western tonal music.

3.2 Composer input and solver output

For chord progressions to be generated, the model requires a series of parameters. These can come from the first layer of the Harmoniser project or from the composer. First, the total number of chords of the musical piece (*n*) and the number of progressions (*l*) must be specified. The progressions' beginning (b_i) and end (e_i) are deduced from the modulations, except for the start of the first progression and the end of the last one, and are known from the start. This is developed in Section 3.4.

$$n, l \in \mathbb{N}_0 \tag{1}$$

$$\forall i \in [0, l[\quad b_i, e_i \in [0, n[\tag{2}$$

$$b_i < e_i \quad b_0 = 0 \quad e_{l-1} = n - 1 \tag{3}$$

The tonality of each progression (t_i), in the form of a tuple (key, mode), must also be provided.

$$\forall i \in [0, l[\quad t_i = (k, m) \tag{4}$$

where $k \in \{\mathrm{C}, \mathrm{C}\sharp, \mathrm{D}, ..., \mathrm{B}\}$ and $m \in \{\text{major, minor}\}$. Different progressions can have the same tonality, but not successively. Additionally, modulation types[3] ($type_m$), starts (s_m) and ends (f_m) must also be specified. Together, they will determine the length of each progression.

$$\forall m \in [0, l-1[$$
$$s_m, f_m \in [0, n[\quad s_m < f_m \tag{5}$$
$$type_m \in \{\text{perfect cadence, pivot chord, alteration, chromatic}\} \tag{6}$$

where m represents a modulation, and is linked to the progression from which it modulates (modulation m goes from progression m to progression $m + 1$).

Provided these parameters, the model gives the chords of the piece. In tonal music, chords are referred to by their degree, i.e. their role in the tonality. However, degrees are tonality specific, and the different progressions must be able to communicate because in the case of modulations, some chords must be constrained by two progressions. This is done using the triplet (root note, quality, state) that uniquely identifies a chord. The model therefore defines three variable arrays for the whole piece R, Q and S that represent each chord's root note, quality and state, as well as an array for the chord degrees in each progression D_i. Additionally, each progression has a subset of the whole piece variable arrays (R_i, Q_i, S_i) that correspond to their part in the piece. These arrays are defined below.

Chord roots The root of a chord is the note on which the chord is built. It is one of the twelve notes (and their enharmonics) of Western music:

$$\forall c \in [0, n[$$
$$R[c] \in \{\text{C, C}\sharp/\text{D}\flat, \text{D, D}\sharp/\text{E}\flat, \text{E/F}\flat, \text{E}\sharp/\text{F F}\sharp/\text{G}\flat, \text{G , G}\sharp/\text{A}\flat, \text{A, A}\sharp/\text{B}\flat, \text{B}\} \tag{7}$$

where c denotes each chord of the progression.

Chord qualities Chord qualities define the intervals of the chord notes with the root of the chord:

$$\forall c \in [0, n[$$
$$Q[c] \in \{\text{Major, Minor, Diminished, Augmented, Dominant seventh, Major seventh, Minor seventh, Diminished seventh, Half-diminished seventh, Minor-major seventh, Augmented sixth}\} \tag{8}$$

Chord states Chord states define the note of the chord that is at the bass, i.e. the lowest note of the chord:

$$\forall c \in [0, n[$$
$$S[c] \in \{\text{Fundamental, First inversion, Second inv., Third inv.}\} \tag{9}$$

Chord degrees The supported chord degrees consist of the seven diatonic chord degrees, as well as some common chromatic chords.

$$\forall i \in [0, l[, \ c_i \in [0, n_i[$$
$$D_i[c_i] \in \{\text{I,II,III,IV,V,VI,VII,Vda, V/II,V/III,V/IV,V/V,V/VI,V/VII,}\flat\text{II, 6}\triangle\} \tag{10}$$

3.3 Progression constraints

In this section and the next we present the most important constraints of the model. We distinguish two categories of constraints: constraints that apply to progressions, i.e. constraints in a given tonality, and constraints that apply to modulations, i.e. between two tonalities. The progression constraints ensure that chord progressions in a tonality follow the rules of tonal harmony, while modulation constraints ensure a smooth transition between the progressions. Due to space limitations, the matrices M, P and L are presented in the technical appendix [4]. In the following definitions, musical notations have been used to present the model more intuitively. In practice, numerical values are used.

[3]Modulation types are detailed in Section 3.4.

[4]http://hdl.handle.net/2078.1/301819

	I	II	III	IV	V	VI	VII	Vda	V/II	V/III	V/IV	V/V	V/VI	V/VII	♭II	6△
I	1	1	1	1	1	1	1		1	1	1	1	1	1	1	1
II	1	1		1	1			1			1	1				
III						1							1			
IV	1	1		1	1		1	1	1			1		1	1	1
V	1			1	1	1					1		1			
VI		1		1	1			1	1		1	1			1	1
VII	1		1							1						
Vda	A				1				B						C	
V/II		1										1				
V/III			1										1			
V/IV				1										1		
V/V					1											
V/VI						1			1							
V/VII	D						1		E	1					F	
♭II					1			1								
6△	G				1			1	H						I	

Table 1: Transition matrix T between successive chord degrees in a tonality. Zeroes are omitted for clarity, thus an empty slot in the matrix corresponds to a value of zero.

Chord transitions The most important constraint describes the possible transitions for chord degrees in a tonality. This is enforced through the T matrix and shown in table 1. It is read as: chord *row* can be followed by chord *column* if the value in the matrix is equal to 1. For example, the III chord can be followed by the VI and V/VI, but not any other degree. The T matrix encodes generally accepted rules for tonal harmony, taken from [Duha, 2016] and [Gauldin, 2004]. Each block of the matrix constrains a specific aspect of tonal chord progressions:

- Block **A** (orange) defines possible chord succession between diatonic chords[5].
- Block **B** (dark blue) defines what secondary dominant[6] chords can follow diatonic chords.
- Block **C** (yellow) defines what chromatic chords, amongst the ones supported, can follow diatonic chords.
- Blocks **D** (green) and **E** (cyan) define what diatonic chords and other secondary dominants can follow secondary dominants, respectively. Secondary dominants must move to a chord that is based on the note that is a perfect fifth below their root note. They can either resolve to their corresponding diatonic chord (e.g. II for V/II), or move to another dominant chord based on that same note (e.g. V/V for V/II).
- Block **F** (grey) enforces the rules for chromatic chords following secondary dominant chords. It is not allowed, so this part of the matrix is empty.
- Blocks **G** (violet) and **H** (magenta) enforce the rules for chromatic chords. The lowered second degree (♭II) and the augmented sixth (6△) must go to V, but they can go to the fifth degree appoggiatura (Vda) before that.
- Block **I** (red) enforces rules for the succession of chromatic chords. It is not allowed, thus this part of the matrix is empty.

T can be seen as an adjacency matrix, thus T^k counts possible chord progressions of k chords in a tonality, which are valid walks through the equivalent graph. Table 2 shows the graph corresponding to the adjacency matrix, separated into the diatonic part (2a) and the chromatic part (2b) for readability. Bold arrows mean that a transition is the preferred choice, regular arrows mean a possible alternative, and dotted arrows mean that a transition is possible but rarely used. Possible transitions through chromatic chords in the diatonic part are annotated on the transition arrow to make the diagram more readable. Since these are nodes, they can be used to hop to the chromatic part and back. The constraint enforced is:

$$\forall i \in [0, l[,\ c'_i \in [0, e_i - b_i[\quad T[D_i[c'_i], D_i[c'_i + 1]] = 1 \quad (11)$$

where i represents each progression and c'_i represents each chord of the progression except the last one, and T is the matrix in Figure 1. This is not implemented with the regular constraint [Pesant, 2004] because using a matrix makes it easy for composers to modify the possible transitions without requiring to recompute the whole underlying DFA of a regular constraint. As explained in Section 4, this does not cause efficiency problems but can be done in the future if it becomes necessary.

Though voicing is handled by *Diatony*, the third layer of the Harmoniser project, a few constraints must be enforced to ensure that the progressions generated by this model are compatible with the strict voicing rules of tonal music, namely tritone resolution and the preparation of diatonic seventh chords. Voicing rules are also necessary for modulations.

Tritone resolution When one of the tritone notes is at the bass, its resolution affects the state of the next chord. This is the case for dominant chords (primary or secondary) in first or third inversion. For chords in first inversion, the bass note should move up by step. For chords in third inversion, it should move down by step.

$$\begin{aligned} &\forall i \in [0, l[,\ c'_i \in [0, e_i - b_i[\\ D = (D_i[c'_i] = \text{V} \wedge Q_i[c'_i] \in \{&\text{Major, Dom. 7th, Dim. 7th}\}) \\ &\vee (\text{V/II} \leq D_i[c'_i] \leq \text{V/VII}) \end{aligned} \quad (12)$$

$$D\ \wedge S_i[c'_i] = 1^{st}\text{inv} \implies B_i[c'_i + 1] = B_i[c'_i] + 1 \mod 7 \quad (13)$$

$$D\ \wedge S_i[c'_i] = 3^{rd}\text{inv} \implies B_i[c'_i + 1] = B_i[c'_i] - 1 \mod 7 \quad (14)$$

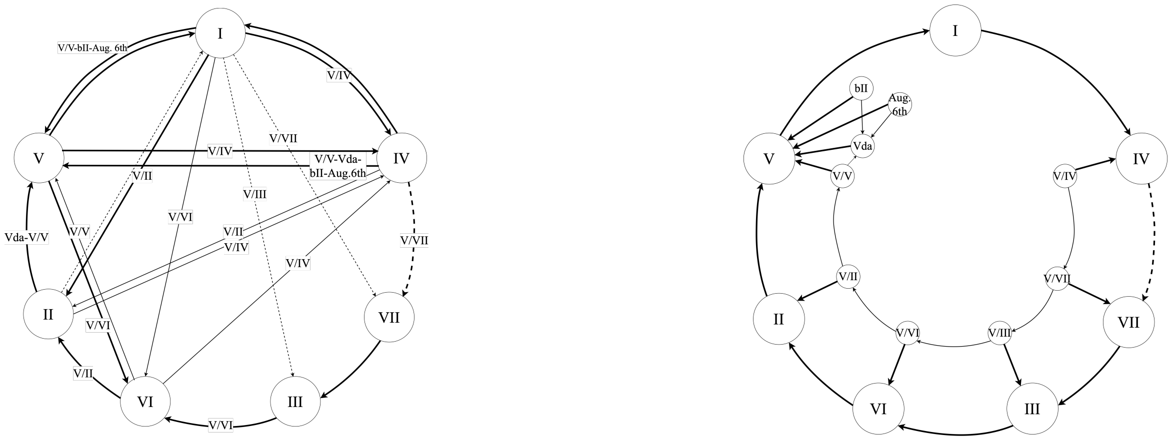
Where D is true for a dominant chord, and false otherwise, and B_i is the array containing the degree of the note at the bass for each chord, which is derived from D_i and S_i and defined in the technical appendix. The " mod 7 " is due to the fact that there are seven diatonic degrees in a tonality.

Preparation of diatonic seventh chords Except for the fifth degree (V) chord, when a diatonic chord has a seventh, that note must be present in the chord that is played before, at the same voice. In our model, we can only enforce that the seventh is in the previous chord. *Diatony* will impose that they are in the same voice.

$$\begin{aligned} &\forall c_i \in [1, e_i - b_i] \\ H_i[c_i] = 1 \wedge D_i[c_i] \leq \text{VII} \wedge D_i[c_i] \neq \text{V} &\implies \\ Ro_i[i-1] = Se_i[i]\ \vee\ Ti_i[i-1] = Se_i[i] & \\ \vee\ Fi_i[i-1] = Se_i[i] & \end{aligned} \quad (15)$$

[5] The fifth degree double appoggiatura (Vda) is treated separately because its musical function is completely different.

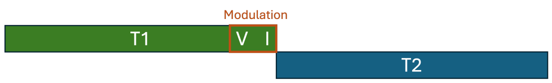
[6] A secondary dominant is the dominant of a diatonic degree.



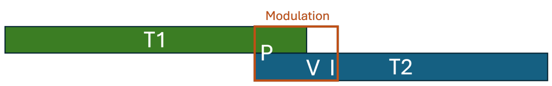
(a) Diatonic part of the graph described by the adjacency matrix in Figure 1.

(b) Chromatic part of the graph described by the adjacency matrix in Figure 1.

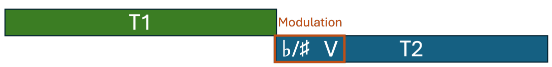
Figure 2: Transition matrix between chord degrees, as a graph. Node names are unique, so walks can hop between (a) and (b).



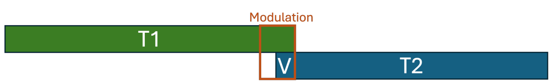
(a) Perfect cadence modulation



(b) Pivot chord modulation



(c) Alteration modulation



(d) Chromatic modulation

Figure 3: Representation of the different modulation types.

Where $H_i[c_i]$ is true when chord c_i in progression i has a seventh, and false otherwise; Ro_i, Ti_i, Fi_i and Se_i are the degree corresponding to the root, third, fifth and seventh of each chord respectively. They are derived from D_i and defined in the technical appendix.

3.4 Modulation constraints

There are two main types of modulation from one tonality to another: modulations to neighbouring tonalities (at least one chord in common), and modulations to distant tonalities. In this paper, we focus on modulations to neighbouring tonalities. We distinguish four types of modulations to neighbouring tonalities. Their representation is given in Figure 3, and their definitions and formalisation are given below.

Perfect cadence modulation This can be considered as one tonality ending and another beginning. The current tonality ends on a perfect cadence and the next tonality starts on the next chord (see Figure 3a). The only constraint to enforce is that the last two chords of the first tonality are V and I, both in fundamental state.

$$\begin{aligned} D_m[e_m - b_m - 1] = \text{V} \ \wedge \ S_m[e_m - b_m - 1] = \text{Fund. State} \\ \wedge \ D_m[e_m - b_m] = \text{I} \ \wedge \ S_m[e_m - b_m] = \text{Fund. State} \end{aligned} \tag{16}$$

We link the first and the second progression to the modulation.

$$e_m = f_m \quad b_{m+1} = f_m + 1 \tag{17}$$

Pivot chord modulation A pivot chord modulation uses a chord that is in both tonalities as a pivot to transition from one tonality to the other. It can be followed by multiple chords that are in both tonalities, and eventually a perfect cadence in the new tonality, which ends the modulation. To model this transition period where chords are in both tonalities, there is an overlap between the two corresponding progressions (see Figure 3b). The global variables from position s_m up to position $f_m - 2$ are constrained by both tonalities, so the chords at these positions must be available in both tonalities. The pivot chord cannot be VII.

$$\begin{aligned} D_{m+1}[f_m - b_{m+1} - 1] = \text{V} \\ \wedge \ S_{m+1}[f_m - b_{m+1} - 1] = \text{Fund. State} \\ \wedge \ D_{m+1}[f_m - b_{m+1}] = \text{I} \\ \wedge \ S_{m+1}[f_m - b_{m+1}] = \text{Fund. State} \end{aligned} \tag{18}$$

We link the first and the second progression to the modulation.

$$e_m = f_m - 2 \quad b_{m+1} = s_m \tag{19}$$

Alteration modulation An alteration modulation introduces a note from the second tonality that is not present in the original tonality to start the modulation. This chord has to be followed by the V of the new tonality, affirming it (see Figure 3c). If the chord used to introduce the alteration cannot be followed by V, it has to be the next chord. The last chord of the first progression must be diatonic, cannot be VII and cannot have a seventh. The first chord of the new progression must be diatonic, and cannot be V or VII.

$$D_m[e_m] \neq \text{VII} \ \wedge \ H_m[e_m] = 0$$
$$\wedge \ D_{m+1}[0] \notin \{\text{V,VII}\} \tag{20}$$
$$D_{m+1}[0] < \text{VII} \wedge D_{m+1}[0] \neq \text{V} \tag{21}$$

Where $H_m[c_m]$ is true when chord c_m has a seventh.

We must also ensure that the first chord of the new progression contains a note that is not in the first tonality. We define a function $f_t(n)$ that takes as argument a note in [C,B], and returns the quality of the diatonic chord built on that note if it is in t. The function is not defined if the note is not in t.

$$f_t(n) = \begin{cases} n \in t & Q_t(n) \\ n \notin t & \perp \end{cases}$$

This is equivalent to a 12-value array, containing for each note the quality of the chord based on this note in t if it exists, and nothing otherwise. We then impose the constraint:

$$f_{t_m}(R_{m+1}[0]) = \perp \ \vee f_{t_{m+1}}(R_{m+1}[0]) \neq f_{t_m}(R_{m+1}[0]) \tag{22}$$

which means that the quality of the chord based on note $R_{m+1}[0]$ cannot be the same in both tonalities. If this note is not in t_m, this is trivially satisfied. This ensures that there is at least one note in the first chord in the new progression that is not in the previous tonality. We still have to enforce that this altered chord is followed by V. Depending on which degree it corresponds to, it might not be possible for V to follow directly. In that case, it should be the next chord.

$$T[D_{m+1}[0], \text{V}] = 0 \implies D_{m+1}[2] = \text{V} \tag{23}$$
$$T[D_{m+1}[0], \text{V}] \neq 0 \implies D_{m+1}[1] = \text{V} \tag{24}$$

Chromatic modulation This kind of modulation occurs when one chord in the first tonality is followed by the V of the new tonality, with a chromatic movement in the voice that plays the leading tone of the new tonality in the dominant chord (see Figure 3d). The voice leading aspect of this modulation needs to be handled in the third (voicing) layer of the Harmoniser project. Similarly to the preparation of diatonic seventh chords, constraints still need to be enforced in this model to make sure that this chromatic movement is possible. In particular, we must enforce that the first chord of the new progression is V, and there must be a one chord overlap between the progressions to ensure that the transition is smooth. The chord in this overlap is thus a secondary dominant in the first tonality, and the dominant in the new one. We must also ensure that the note in the first tonality corresponding to the leading tone in the new tonality is present in the chord just before the dominant of the new tonality (i.e., when modulating from C major to A major, there must be a G in the first chord that can move to a G♯ in the second chord). To enforce that, we must compute the interval in semitones between the keys of the two tonalities, and transform that into a degree difference. This is shown in Table 2.

$$d = Degs[|t_m.k_m - t_{m+1}.k_{m+1}|]$$
$$s = 6 + d \mod 7$$
$$D_{m+1}[0] = V \ \wedge \ Ro_m[n_m - 2] = s$$
$$\vee \ Ti_m[n_m - 2] = s \vee Fi_m[n_m - 2] = s \tag{25}$$

Where s is the degree that the seventh of the new tonality corresponds to in the first tonality.

3.5 Branching

The goal of our model is to define a search space that is as permissive as possible, only enforcing mandatory rules of tonal harmony to allow for composers' creativity to shape the solutions instead of the constraints. As a result, the number of solutions is very large and the branching strategies are defined for the relevancy of solutions rather than for efficiency.

With that in mind, the branching is first performed on chord degrees, as this is the most important variable array, selecting the variable with the smallest domain size and the value at random. The preferences in Figure 2 are not followed to avoid staying in the "preferred" transitions that would be repetitive. This could of course be improved in the future by considering composer preferences when assigning new values to variables. Branching is then performed on states, also on the smallest domain variable, favouring fundamental state and first inversion as these are the most common states in tonal music. Finally, branching is performed on chord qualities, favouring triads over seventh chords.

4 Evaluation and example use case

Complete source code of our model is available on GitHub[7], along with its integration with Diatony.

4.1 Efficiency and number of solutions

Since the model is designed to give as much freedom as possible to composers, the number of possible solutions for a given problem is enormous if no composer preferences are given. The only constraints enforced by default are the ones that are necessary to ensure that the generated chord progressions follow the rules of tonal harmony. We expect the composer to add musical ideas to guide the solver, formulated as constraints. This will in the future be done through a GUI.

As a result of this approach, solutions are found extremely quickly for problems of significant size and efficiency is hence not the main focus of this section. For example, the musical piece shown in Section 4.2 was generated in 3ms on an M1 MacBook Pro. Another longer piece, consisting of 60 chords with five modulations, was generated in 20ms. This is because this layer of the Harmoniser project on its own lacks global rules, that will be enforced through the first layer in future work. We expect the computation to be more intensive with the addition of composer-subjective constraints, that will transform the problem from a satisfaction problem (finding a valid solution) to an optimisation problem (finding the best solution), where the criteria for what makes a solution better are provided by the composer, and with complex links between the progressions.

4.2 Example of composer use

We now put ourselves in the mindset of a composer, to show how our tool can be used to generate a harmonic progression.

[7]https://github.com/sprockeelsd/Progressions-and-Modulations/tree/IJCAI2025

0	1	2	3	4	5	6	7	8	9	10	11
unison (0)	second (1)		third (2)		fourth (3)	fifth (4)		sixth (5)		seventh (6)	

Table 2: Conversion between intervals and degree difference (Degs). The first row correspond to intervals in semitones.

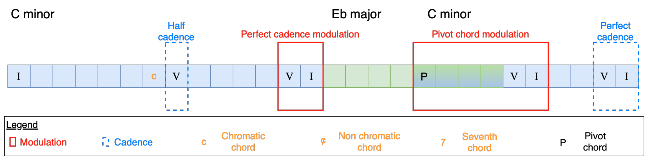


Figure 4: Representation of the input given to the solver.

We explain the example in terms of constraints, but in a practical tool, these constraints would be given through a GUI. For our example, we want to write a chord progression that starts in C minor, modulates to its relative tonality E♭ major and then comes back to the tonality of C minor, that is 28 chords long. We want a perfect cadence modulation on chord 12, and a pivot chord modulation from chord 18 to 23.

In addition to these necessary instructions, we add the following constraints to express our musical intentions. (1) The first chord must be I. (2) No chord can be III or VII, except during the pivot chord modulation. (3) Only dominant chords can have a seventh. (4) There is a half cadence on chord seven. (5) There is a chromatic chord just before that (position six). (6) The II chord should only be used in first inversion. (7) There is a perfect cadence at the end of the piece. The input is illustrated in Figure 4.

If we run the solver with these input and constraints, it produces the following output. For the first progression in C minor, the chords suggested by the solver are *I-II-Vda-V-VI-IV-♭II-V-I-V-I-IV-V-I*. This is interesting, but based on personal taste, we make a few modifications: *I-II-Vda-V-I-IV-6△-V-I-V-I-Vda-V-I*. This is also an accepted solution for the solver. For the E♭ major to C minor progression, the output is *I-VI-II-V-VI|I-V/VI|V-VI|I-VII|II-V-I-VI-II-V-I*, where *VI|I* is the pivot chord starting the modulation (sixth degree in E♭ major and first degree in C minor) and the following chords are in both tonalities up until the perfect cadence in C minor. For this part, we only make one small modification: we modify the perfect cadence that ends the modulation to be an interrupted cadence. This is due to personal taste. The final chords for this part are thus *I-VI-II-V-VI|I-V/VI|V-VI|I-VII|II-V-VI-I-II-V-I*. Chord states have been omitted in the listing of the output to keep it readable, but they are as shown in Figure 5. *Diatony* can then be used to generate a four-voice texture representing our piece. Figure 5 shows one possible four-voice texture of this piece. It can be listened to here[8]. A harmonic rhythm has been given to the chords, as well as some ornamental notes, by the composer.

5 Conclusion

This paper defines a formal model of tonal chord progressions and modulations to neighbouring tonalities. We give a constraint-based implementation of this model in the Gecode constraint solver. Combined with the *Diatony* model [Sprockeels and Van Roy, 2024], this implementation generates tonal chord progressions with modulations in a four-voice texture.

Figure 5: Musical piece based on the chord progression generated by the solver. The voicing has been added by Diatony and the rhythm by the composer.

The present model can be used as the foundation for many useful extensions. It can be enriched by allowing for more chromatic and borrowed chords, as well as modulations to distant tonalities, and by adding larger harmonic structures like harmonic sequences. The matrices encoding large amounts of musical knowledge, such as T and others defined in the technical appendix, are currently encoded in Gecode by element constraints, and could be extended to give a weight to each value, allowing composers to value some choices more than others. Extensional constraints could also be used instead of constant values, to dynamically change values during the search. Global constraints such as the regular or cost-regular constraints could also be used to further improve the efficiency of the model.

In the case where the composer wants suggestions from the solver, it would be interesting to generate successive solutions that differ significantly. This could be done using a branch and bound approach to post additional constraints when a solution is found, or using other approaches such as those proposed in [Pesant *et al.*, 2022] and [Ingmar *et al.*, 2020]. This is left for future work.

This work is part of the ongoing Harmoniser project aiming to assist composers in their creation process with constraint programming. In this project, a four-layer decomposition of the composition process was identified. So far, models have been defined and implemented for layer 2 (this paper) and layer 3 ([Sprockeels and Van Roy, 2024]). Models for the remaining layers are ongoing work with the goal of providing a complete set of models for the whole composition process, allowing to generate full musical pieces with the help of constraint programming. We are also working on a graphical user interface for this tool as well as an implementation as a plug-in for a Digital Audio Workstation.

[8] https://youtu.be/97wBAwcZC8E?si=o_dvYZyOGCegtrHN

Acknowledgements

The authors wish to thank the Conservatoire Royal de Bruxelles for allowing us to take courses on music theory, and more specifically we thank Prof. Adrien Tsilogiannis for his insight on this paper, as well as Dr. Karim Haddad from IRCAM for his insight on the musical aspect of the paper. We also thank Juliette Vanderhaeghen and Lucile Dierckx for their support and feedback during the redaction. We thank all the anonymous reviewers of this paper who helped us to greatly improve the quality of the paper.

References

[Anders and Miranda, 2009] Torsten Anders and Eduardo R Miranda. A Computational Model that Generalises Schoenberg's Guidelines for Favourable Chord Progressions. In *proceedings of the Sound and Music Computing Conference*, pages 48–52, 2009.

[Anders *et al.*, 2005] Torsten Anders, Christina Anagnostopoulou, and Michael Alcorn. Strasheela: Design and Usage of a Music Composition Environment Based on the Oz Programming Model. In *Multiparadigm Programming in Mozart/Oz: Second International Conference, MOZ 2004, Charleroi, Belgium, October 7-8, 2004, Revised Selected and Invited Papers 2*, pages 277–291. Springer, 2005.

[Anders, 2008] Torsten Anders. *Composing Music by Composing Rules: Design and Usage of a Generic Music Constraint System*. PhD thesis, Queen's University Belfast, 2008.

[Bonlarron and Régin, 2024] Alexandre Bonlarron and Jean-Charles Régin. Intertwining CP and NLP: The Generation of Unreasonably Constrained Sentences. In *Thirty-Third International Joint Conference on Artificial Intelligence* {*IJCAI-24*}, pages 7600–7608. International Joint Conferences on Artificial Intelligence Organization, 2024.

[Carpentier *et al.*, 2010] Grégoire Carpentier, Gérard Assayag, and Emmanuel Saint-James. Solving the Musical Orchestration Problem using Multiobjective Constrained Optimization with a Genetic Local Search Approach. *Journal of Heuristics*, 16:681–714, 2010.

[Davismoon and Eccles, 2010] Stephen Davismoon and John Eccles. Combining Musical Constraints with Markov Transition Probabilities to Improve the Generation of Creative Musical Structures. In *European Conference on the Applications of Evolutionary Computation*, pages 361–370. Springer, 2010.

[Duha, 2016] Isabelle Duha. *L'Harmonie en Liberté: de la Mémoire à l'Improvisation*. Gérard Billaudot, Armiane Imp., 2016.

[Ebcioğlu, 1990] Kemal Ebcioğlu. An Expert System for Harmonizing Chorales in the Style of JS Bach. *The Journal of Logic Programming*, 8(1-2):145–185, 1990.

[Eppe *et al.*, 2015] Manfred Eppe, Roberto Confalonieri, Ewen Maclean, Maximos Kaliakatsos, Emilios Cambouropoulos, Marco Schorlemmer, Mihai Codescu, and K Kühnberger. Computational Invention of Cadences and Chord Progressions by Conceptual Chord-Blending. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 2445–2451. AAAI Press; International Joint Conferences on Artificial Intelligence, 2015.

[Gauldin, 2004] Robert Gauldin. *Harmonic Practice in Tonal Music. Second Edition*. W. W. Norton and Company, Inc, 2004.

[Gecode Team, 2019] Gecode Team. *Gecode: Generic Constraint Development Environment*, 2019.

[Giuliani *et al.*, 2023] Luca Giuliani, Francesco Ballerini, Allegra De Filippo, and Andrea Borghesi. MusiComb: a Sample-based Approach to Music Generation Through Constraints. In *2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 194–198, 2023.

[Herremans and Sörensen, 2013] Dorien Herremans and Kenneth Sörensen. Composing Fifth Species Counterpoint Music with a Variable Neighborhood Search Algorithm. *Expert systems with applications*, 40(16):6427–6437, 2013.

[Huang and Chew, 2005] Cheng Zhi Anna Huang and Elaine Chew. Palestrina Pal: a Grammar Checker for Music Compositions in the Style of Palestrina. In *Proceedings of the 5th Conference on Understanding and Creating Music*, 2005.

[Ingmar *et al.*, 2020] Linnea Ingmar, Maria Garcia de la Banda, Peter J Stuckey, and Guido Tack. Modelling Diversity of Solutions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1528–1535, 2020.

[Lattner *et al.*, 2018] Stefan Lattner, Maarten Grachten, and Gerhard Widmer. Imposing Higher-level Structure in Polyphonic Music Generation using Convolutional Restricted Boltzmann Machines and Constraints. *Journal of Creative Music Systems*, 2:[1]–31, 2018.

[Laurson and Kuuskankare, 2005] Mikael Laurson and Mika Kuuskankare. Extensible Constraint Syntax through Score Accessors. In *Journées d'Informatique Musicale*, 2005.

[Pachet and Roy, 2001] François Pachet and Pierre Roy. Musical Harmonization with Constraints: A Survey. *Constraints*, 6:7–19, 2001.

[Pachet and Roy, 2011] François Pachet and Pierre Roy. Markov Constraints: Steerable Generation of Markov Sequences. *Constraints*, 16(2):148–172, 2011.

[Papadopoulos *et al.*, 2015] Alexandre Papadopoulos, Pierre Roy, Jean-Charles Régin, and François Pachet. Generating all Possible Palindromes from n-gram Corpora. In *IJCAI 2015*, 2015.

[Pesant *et al.*, 2022] Gilles Pesant, Claude-Guy Quimper, and Hélène Verhaeghe. Practically Uniform Solution Sampling in Constraint Programming. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 335–344. Springer, 2022.

[Pesant, 2004] Gilles Pesant. A Regular Language Membership Constraint for Finite Sequences of Variables. In *International conference on principles and practice of constraint programming*, pages 482–495. Springer, 2004.

[Rohrmeier, 2011] Martin Rohrmeier. Towards a Generative Syntax of Tonal Harmony. *Journal of Mathematics and Music*, 5(1):35–53, 2011.

[Sandred, 2010] Örjan Sandred. PWMC, a Constraint-solving System for Generating Music Scores. *Computer Music Journal*, 34(2):8–24, 2010.

[Sprockeels and Van Roy, 2024] Damien Sprockeels and Peter Van Roy. Expressing Musical Ideas with Constraint Programming Using a Model of Tonal Harmony. In *International Joint Conference on Artificial Intelligence*, 2024.

[Truchet *et al.*, 2003] Charlotte Truchet, Gérard Assayag, and Philippe Codognet. OMClouds, Petits Nuages de Contrainte dans OpenMusic. In *Journées d'Informatique Musicale*, 2003.

Towards a Practical Tool for Music Composition: Using Constraint Programming to Model Chord Progressions and Modulations: Technical Appendix

Damien Sprockeels , **Peter Van Roy**
ICTEAM, UCLouvain, Louvain-la-Neuve, Belgium
{damien.sprockeels, peter.vanroy}@uclouvain.be

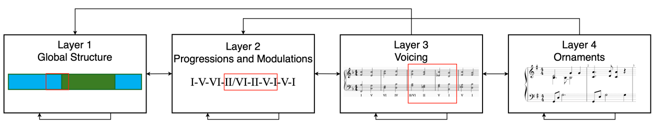


Figure 1: Illustration of the four-layer framework for composing a musical piece.

1 Introduction

This document defines the mathematical formalization of the second layer of the *Harmoniser project*, namely chord progressions and modulations in the context of tonal harmony.

2 Structure of Harmoniser

The Harmoniser project aims to build a practical tool to aid composers based on CP that addresses both the issues of rule definition and computation time. Because the solver enforces rules, this relieves the composer of much tedious work so they can focus on adding musical ideas to shape solutions into a desired result. To reduce computational complexity, we follow the decomposition of the composition process that is detailed in the paper. It decomposes the process of musical composition in four layers, resulting in smaller problems that can be solved independently. This is depicted in Figure 1. The first layer is global harmonic structure, the second is chord progressions and modulations, the third is chord voicing, and the fourth adds ornaments. The present paper focuses on the second layer. The third layer's model has already been developed (see full paper for the reference). The model presented in this annex is the second layer. Its output can be given to the third layer, which generates the four voices and hence the notes.

3 Composer Input

This section describes the parameters that are required to define instances of the problem. On top of that, the composer can add as many constraints as they want to shape the results.

Whole piece Two parameters are defined regarding the whole piece: the total number of chords, $n \in \mathbb{N}_0$, and the number of progressions, $l \in \mathbb{N}_0$.

Progressions Three parameters are required to define a progression: its beginning b_i, end e_i and tonality t_i. The beginning and end are deduced from the modulations (see Section 7.2).

$$\forall i \in [0, l[$$

$$b_i, e_i \in [0, n[\quad b_0 = 0 \quad e_{l-1} = n - 1 \tag{1}$$

$$t_i = (k, m) \tag{2}$$

where $k \in$ {C, C♯/D♭, D, D♯/E♭, E/F♭, E♯/F, F♯/G♭, G, G♯/A♭, A, A♯/B♭, B} and $m \in$ {major, minor}.

Modulations Finally, modulations are also defined with three parameters: their start s_m, finish f_m and type $type_m$.

$$\forall m \in [0, l-1[$$

$$s_m, f_m \in [0, n[\quad s_m < f_m \tag{3}$$

$$type_m \in \{\text{perfect cadence, pivot chord, alteration, chromatic}\} \tag{4}$$

where m represents a modulation, and is linked to the progression from which it modulates (modulation m goes from progression m to progression $m + 1$).

4 Solver Output

Given the parameters described in section 3, the solver will produce the following triplet as an output:

- The array of root notes for each chord of the whole piece.
- The array of states for each chord of the whole piece.
- The array of chord qualities for each chord of the whole piece.

To make the output easier to analyze in the context of tonal harmony, additional arrays are generated for each progression in the piece expressing the chords as degrees in the respective tonality. the progressions' beginning and end, deduced from the modulations, are also generated. This makes it possible to give the solver's output directly as an input to layer 3 of the Harmoniser project.

5 Indices in the whole piece

This section defines useful indices to access elements in the whole piece.

Fundamental State	0
First Inversion	1
Second Inversion	2
Third Inversion	3
Fourth Inversion	4

Table 1: Value table for the possible chord states

Chord position in the piece As the model writes a musical piece as a sequence of chords, it is useful to be able to access each of them. This is done through their index:

$$c \in [0, n[\tag{5}$$

A number of rules apply on all chords but the last, so we define an index for that as well:

$$c' \in [0, n-1[\tag{6}$$

Progressions As there can be multiple progressions (connected by modulations) in the musical piece, it is useful to be able to access each of them. This is done through their index:

$$p \in [0, l[\tag{7}$$

Chord position in a section As explained in section 2, the piece is divided in progressions based on the tonalities. It is also useful to be able to access a chord in these sections, so we define an index for them as well:

$$\forall p \quad c_p \in [0, n_p - 1] \tag{8}$$

where n_p is the number of chord in the progression p. Similarly,

$$\forall p \quad c'_p \in [0, n_p - 2] \tag{9}$$

6 Variables

This section defines the variables used to express the musical rules. Some variables are global for the whole piece, while others are specific to a progression.

6.1 Global variables

States This array of variables contains the state of each chord.

$$\forall c \quad S[c] \in \{\text{Fundamental, First inversion, Second inv., Third inv., Fourth inv.}\} \tag{10}$$

The corresponding values are given in Table 1.

Qualities This array of variables contains the quality of the chords.

$$\forall c \quad Q[c] \in \{\text{Major (M), Minor (m), Diminished (°), Augmented (}\triangle\text{), Dominant seventh (7), Major seventh (M7), Minor seventh (m7), Diminished seventh (°7), Half-diminished seventh (Ø7), Minor-major seventh (mM7), Augmented sixth (6}\triangle\text{)}\} \tag{11}$$

Major	0	Minor	1
Diminished	2	Augmented	3
Dominant Seventh	4	Major Seventh	5
Minor Seventh	6	Diminished Seventh	7
Minor Major Seventh	8	Augmented Sixth	9

Table 2: Value table for the different chord qualities

	Third	Fifth	Seventh
Major	4	3	
Minor	3	4	
Diminished	3	3	
Augmented	4	4	
Augmented sixth	4	6	
Dominant seventh	4	3	3
Major seventh	4	3	4
Minor seventh	3	4	3
Diminished seventh	3	3	3
Half diminished	3	3	4
Minor major seventh	3	4	4

Table 3: Interval (in semitones) with the previous note for each chord quality

The corresponding values are given in Table 2[1]. Table 3 gives the intervals between each consecutive note of the chord, based on its quality. Together with the root note, it allows to select all the notes constituting the chord.

Root Notes This array of variables contains the root note of the chord, i.e the note it is built on.

$$\forall c \quad R[c] \in \{\text{C, C}\sharp\text{/D}\flat\text{, D, D}\sharp\text{/E}\flat\text{, E/F}\flat\text{, E}\sharp\text{/F, F}\sharp\text{/G}\flat\text{, G , G}\sharp\text{/A}\flat\text{, A, A}\sharp\text{/B}\flat\text{, B}\} \tag{12}$$

The corresponding values are given in Table 4.

Has Seventh This array of variables contains 1 for chords that have a seventh, and 0 for chords that don't. A chord can have a seventh on top of the traditional third and fifth. In that case, some extra rules must be enforced, so it is important to have an easy access to that information. We thus define an array H:

$$\forall c \, H[c] \in [0, 1] \tag{13}$$

Where 1 means that the chord has a seventh, and 0 means it doesn't. It is linked to the quality array:

$$\forall c \, H[c] = 0 \Leftrightarrow Q[c] \leq 3 \tag{14}$$

$$\forall c \, H[c] = 1 \Leftrightarrow Q[c] > 3 \tag{15}$$

6.2 Variables specific to a progression

This section defines the variables defined for each progression.

[1] Only the Italian augmented sixth is implemented, but other versions of these chords are straightforward to include in the model.

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

Table 4: Value table for the different notes

I	II	III	IV	V	VI	VII	Vda
0	1	2	3	4	5	6	7
V/II	V/III	V/IV	V/V	V/VI	VII°	♭ II	6 △
8	9	10	11	12	13	14	15

Table 5: Value table for the different notes

Chords This is the main array of variables for the chord progression. It contains the degree with respect to the tonality for each chord in this progression.

$$\forall p,\ \forall c_p$$
$$D_p[c_p] \in \{\text{I,II,III,IV,V,VI,VII,Vda}, \text{V/II,V/III,V/IV,V/V,V/VI,V/VII,♭II}, 6\triangle\} \tag{16}$$

The corresponding values are given in Table 5.

States This array contains the state of each chord. It is linked to the global array of states through the following formula:

$$\forall p \quad S_p[0 : n_p - 1] \equiv S[b_p : e_p] \tag{17}$$

Qualities This array contains the quality of each chord. It is linked to the global array of qualities through the following formula:

$$\forall p \quad Q_p[0 : n_p - 1] \equiv Q[b_p : e_p] \tag{18}$$

Root Notes This array contains the root note of each chord. It is linked to the global array of root notes through the following formula:

$$\forall p \quad R_p[0 : n_p - 1] \equiv R[b_p : e_p] \tag{19}$$

Bass Degrees This is the note at the bass in a broad sense, i.e without taking the alteration into account. For example, if the tonality is C major and the bass degree is III, the note can be E, E♭ or E♯. There are seven degrees in a tonality.

$$B_p[c_p] \in \{\text{I,II,III,IV,V,VI,VII}\} \tag{20}$$

Root note, third, fifth and seventh In tonal music, chords are built by stacking notes that are a third apart. For example, the I chord is built on the I, then has the III (a third above), then the V (a third above III), and so on. Table 11 gives the degrees present in each chord degree. This is useful to determine what chord degree is at the bass of a chord depending on its state, because the state is determined by the note at the bass and vice versa and constraints have to be posted on the bass.

$$Ro_p[c_p] \in \{\text{I,II,III,IV,V,VI,VII}\} \tag{21}$$
$$Ti_p[c_p] \in \{\text{I,II,III,IV,V,VI,VII}\} \tag{22}$$
$$Fi_p[c_p] \in \{\text{I,II,III,IV,V,VI,VII}\} \tag{23}$$
$$Se_p[c_p] \in \{\text{I,II,III,IV,V,VI,VII}\} \tag{24}$$

	I	II	III	IV	V	VI	VII	Vda	V/II	V/III	V/IV	V/V	V/VI	V/VII	♭II	6△
I	1	1	1	1	1	1	1		1	1	1	1	1	1	1	1
II	1	1		1	1			1			1	1				
III						1							1			
IV	1	1		1	1		1	1	1			1		1	1	1
V	1			1	1	1					1		1			
VI		1		1	1			1	1		1	1			1	1
VII	1		1							1						
Vda	A				1				B						C	
V/II		1										1				
V/III			1										1			
V/IV				1										1		
V/V					1											
V/VI						1			1							
V/VII	D						1		E	1					F	
♭II					1			1								
6△	G				1			1	H						I	

Table 6: Transition matrix T between successive chord degrees in a tonality. Zeroes are omitted for clarity, thus an empty slot in the matrix corresponds to a value of zero.

Is Chromatic A chord is considered chromatic in a progression if it contains at least one note that does not occur naturally in that progression's tonality. In Table 5, this corresponds to any chord in the second row. Therefore, the definition of this array is:

$$\forall c_p\ I_p[c_p] \in [0, 1] \tag{25}$$

The following equations give the values:

$$\forall c_p\ I_p[c_p] = 0 \Leftrightarrow D_p[c_p] \leq \text{Vda} \tag{26}$$
$$\forall c_p\ I_p[c_p] = 1 \Leftrightarrow D_p[c_p] > \text{Vda} \tag{27}$$

Has Seventh This array contains 1 if the chord has a seventh, and 0 if it doesn't. It is linked to the global array through the following formula:

$$\forall p \quad H_p[0 : n_p - 1] \equiv H[b_p : e_p] \tag{28}$$

7 Constraints

This section defines all the constraints of the model.

7.1 Progression constraints

This section defines all the constraints regarding chord progressions in a given tonality.

Possible chord transitions The most important constraint is the one that defines what chord degrees can be reached from a given degree. This is done using the matrix depicted in Table 6:

$$\forall p, c'_p \quad T[D_p[c'_p]][D_p[c'_p + 1]] = 1 \tag{29}$$

Each row and column represents a possible chord in the tonality. Each row contains a 1 for chords that are reachable from this chord, and a 0 for chords that are not reachable. This matrix is not tonality specific. For more details regarding the matrix, see the main part of paper.

Link between chord degrees and qualities Each degree has a set of chord qualities that are possible. These depend on the mode of the tonality (major/minor). This is enforced through the following constraint:

$$\forall p, c_p \quad M_{t_p.m}[D_p[c_p]][Q_p[c_p]] = 1 \tag{30}$$

where M can either be M_0 (Figure 7) or M_1 (Figure 8) depending on the mode of the tonality of the progression.

	M	m	°	△	7	M7	m7	°7	Ø7	mM7	6△
I	1					1					
II		1					1				
III		1					1				
IV	1					1					
V	1				1			1			
VI		1					1				
VII			1						1		
Vda	1										
V/II	1				1			1			
V/III	1				1			1			
V/IV	1				1			1			
V/V	1				1			1			
V/VI	1				1			1			
V/VII	1				1			1			
♭II	1										
6△											1

Table 7: Matrix linking the chord degrees with the possible qualities for major tonalities (M_0)

	M	m	°	△	7	M7	m7	°7	Ø7	mM7	6△
I		1					1				
II			1						1		
III	1					1					
IV		1					1				
V	1				1			1			
VI	1					1					
VII			1					1			
Vda		1									
V/II	1				1			1			
V/III	1				1			1			
V/IV	1				1			1			
V/V	1				1			1			
V/VI	1				1			1			
V/VII	1				1			1			
♭II	1										
6△											1

Table 8: Matrix linking the chord degrees with the possible qualities for minor tonalities (M_1)

Link between chord degrees and states Similarly, each chord degree has a possible set of states:

$$\forall p, c_p \quad P[D_p[c_p]][S_p[c_p]] = 1 \tag{31}$$

where P is depicted in Figure 9. This is independent of the tonality.

Link between chord degrees and root notes Again, similarly, each chord degree corresponds to a note value:

$$\forall p, c_p \quad R_p[c_p] = N[t_p, D_p[c_p]] \tag{32}$$

Where N is a matrix where each row represents a tonality (defined by key and mode), and each column corresponds to a degree within that tonality (as defined in Table 5). The entries of N contain the corresponding note values (as values from Table 4) for each degree in each tonality. Table 10 shows the entry of the N matrix for the G major tonality. The full matrix is omitted due to its size and the fact that it can be easily reconstructed, as all values are systematically derived from the definitions of tonal music.

Link between the chord degrees, bass degrees and states The degree that is present at the bass for a given chord in a given state is given by Table 11:

$$B_p[c_p] = L[D_p[c_p]][S_p[c_p]] \tag{33}$$

Root, third, fifth and seventh of a chord Similarly, Table 11 gives the degree corresponding to the root, third, fifth and

Possible states per chord degree P																
	I	II	III	IV	V	VI	VII	Vda	V/II	V/III	V/IV	V/V	V/VI	V/VII	♭II	6△
Fund.	1	1	1	1	1	1	1		1	1	1	1	1	1	1	1
1^{st}	1	1		1	1		1		1	1	1	1	1	1	1	
2^{nd}					1		1	1	1	1	1	1	1	1		
3^{rd}					1				1	1	1	1	1	1		
4^{th}																

Table 9: Matrix of possible states per degree (P)

I	II	III	IV	V	VI	VII	Vda
G	A	B	C	D	E	F♯	G

V/II	V/III	V/IV	V/V	V/VI	V/VII	♭ II	6 △
E	F♯	G	A	B	C♯	A♭	E♭

Table 10: N matrix for the tonality of G major

seventh of a chord:

$$Ro_p[c_p] = L[D_p[c_p], 0] \tag{34}$$
$$Ti_p[c_p] = L[D_p[c_p], 1] \tag{35}$$
$$Fi_p[c_p] = L[D_p[c_p], 2] \tag{36}$$
$$Se_p[c_p] = L[D_p[c_p], 3] \tag{37}$$

Fifth degree appoggiatura The Vda chord must resolve to the V. This is already enforced with the matrix *T* from Table 6. The V chord should be in fundamental state, and the quality should be either major or dominant seventh.

$$\begin{aligned} &\forall p, c'_p \\ D_p[c'_p] = \text{Vda} \implies &S_p[c'_p + 1] = \text{Fund. state} \\ &\wedge Q_p[c'_p + 1] \in \{\text{Major, Dominant seventh}\} \end{aligned} \tag{38}$$

Neapolitan sixth The chord based on the flattened second degree (♭II, one semitone above the tonic) should be used in first inversion.

$$\forall p, c_p \quad D_p[c_p] = \flat\text{II} \implies S_p[c_p] = \text{First inv.} \tag{39}$$

Successive chords of same degree This rule is not from a music treatise, but is due to the representation choices we made. Since ornaments and melody are not considered here, a chord that varies over time but stays in the same state is considered as one chord. It follows that two successive chords of the same degree cannot have the same quality and state.

$$\begin{aligned} \forall p, c'_p \quad D_p[c'_p] = D_p[c'_p + 1] \implies \\ S_p[c'_p] \neq S_p[c'_p + 1] \vee Q_p[c'_p] \neq Q_p[c'_p + 1] \end{aligned} \tag{40}$$

Tritone resolutions One aspect of voice leading that we do have to take into consideration in some cases is the tritone resolution. In particular, when one of the tritone notes is at the bass, it affects the state of the next chord. This is the case for dominant chords (primary or secondary) in first or third inversion. For chords in first inversion, the bass note should move up by step. For chords in third inversion, it should move

Chord composition L																
	I	II	III	IV	V	VI	VII	Vda	V/II	V/III	V/IV	V/V	V/VI	V/VII	♭II	6△
Fund./root	I	II	III	IV	V	VI	VII	I	VI	VII	I	II	III	VII	II	VI
1^{st}/third	III	IV	V	VI	VII	I	II	III	I	II	III	IV	V	II	IV	I
2^{nd}/fifth	V	VI	VII	I	II	III	IV	V	III	IV	V	VI	VII	IV	VI	IV
3^{rd}/seventh	VII	I	II	III	IV	V	VI	VII	V	VI	VII	I	II	VI	I	-

Table 11: Matrix stating the note degrees that compose each chord degree.

down by step.

$$\begin{aligned}\forall p, c'_p\ D = (D_p[c'_p] = \text{V} \wedge Q_p[c'_p] \in \\ \{\text{Major, Dominant Seventh, Diminished seventh}\}) \\ \vee \left(\text{V/II} \leq D_p[c'_p] \leq \text{V/VII}\right) \\ D\ \wedge\ S_p[c'_p] = \text{First inv.} \implies \\ B_p[c'_p + 1] = B_p[c'_p] + 1 \mod 7 \\ D\ \wedge\ S_p[c'_p] = \text{Third inv.} \implies \\ B_p[c'_p + 1] = B_p[c'_i] - 1 \mod 7 \end{aligned} \tag{41}$$

Where D is true for a dominant chord (V with a major, dominant seventh or diminished seventh quality or secondary dominant chord), and false otherwise.

Third inversion Chords that have no seventh cannot be in third inversion.

$$\begin{aligned}\forall p, c_p \\ S[c_p] = \text{Third inv.} \implies Q[c_p] \geq \text{Dominant Seventh}\end{aligned} \tag{42}$$

The seventh must be prepared Aside from dominant seventh chords, the seventh of diatonic chords must be present in the chord that is played before the chord containing the seventh, at the same voice. In our model, we only need to ensure that the seventh is in the previous chord.

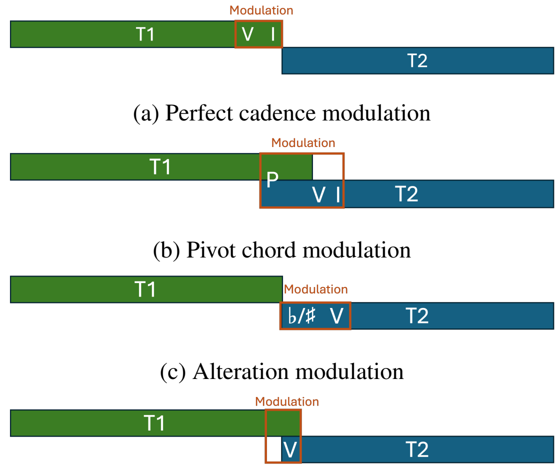
$$\begin{aligned}\forall c_p \in [1, n_p - 1] \\ H_p[c_p] = 1 \wedge Q_p[c_p] \neq \text{Dom. Seventh} \wedge D_p[c_p] \leq \text{VII} \\ \implies Ro[p-1] = Se[p]\ \vee\ Ti[p-1] = Se[p] \\ \vee\ Fi[p-1] = Se[p]\end{aligned} \tag{43}$$

Secondary dominant of the seventh degree This chord is only available in the minor mode, as the root note of this chord is not diatonic in the major mode.

$$\forall p, c_p \quad t_p.m_p = \text{Major} \implies D_p[c_p] \neq \text{V/VII} \tag{44}$$

Diminished seventh dominant chords These chords are special because they cannot be in fundamental state, since they do not have their fundamental as one of their notes. Indeed, this chord is formed by adding a minor third on top of a dominant seventh chord, and by removing its fundamental. For example, in C major, the dominant seventh chord is *G-B-D-F*, and the diminished seventh version is *B-D-F-A*♭. It is thus considered to be in first inversion, so the note at the bass is correct.

$$\begin{aligned}\forall p, c_p \quad Q_p[c_p] = \text{Dim. seventh} \wedge D_p[c_p] \neq \text{VII} \implies \\ S_p[c_p] \neq \text{Fund. state}\end{aligned} \tag{45}$$



(a) Perfect cadence modulation

(b) Pivot chord modulation

(c) Alteration modulation

(d) Chromatic modulation

Figure 2: Representation of the different modulation types.

7.2 Modulation constraints

There are two main types of modulation from one tonality to another: modulations to neighbouring tonalities (at least one chord in common), and modulations to distant tonalities. In this paper, we focus on modulations to neighbouring tonalities. We distinguish four types of modulations to neighbouring tonalities. Their representation is given in Figure 2, and their definitions and formalisation are given below.

Perfect cadence modulation This can be considered as one tonality ending and another beginning. The current tonality ends on a perfect cadence and the next tonality starts on the next chord (see Figure 2a). The only constraint to enforce is that the last two chords of the first tonality are V and I, both in fundamental state.

$$\begin{aligned}D_m[e_m - b_m - 1] = \text{V}\ \wedge\ S_m[e_m - b_m - 1] = \text{Fund. State} \\ \wedge\ D_m[e_m - b_m] = \text{I}\ \wedge\ S_m[e_m - b_m] = \text{Fund. State}\end{aligned} \tag{46}$$

We link the first and the second progression to the modulation.

$$e_m = f_m \quad b_{m+1} = f_m + 1 \tag{47}$$

Pivot chord modulation A pivot chord modulation uses a chord that is in both tonalities as a pivot to transition from one tonality to the other. It can be followed by multiple chords that are in both tonalities, and eventually a perfect cadence in the new tonality, which ends the modulation. To model this transition period where chords are in both tonalities, there is an overlap between the two corresponding progressions (see Figure 2b). The global variables from position s_m up to position $f_m - 2$ are constrained by both tonalities, so the chords at these positions must be available in both tonalities. The pivot

0	1	2	3	4	5	6	7	8	9	10	11
unison (0)	second (1)		third (2)		fourth (3)	fifth (4)		sixth (5)		seventh (6)	

Table 12: Conversion between intervals and degree difference (Degs). The first row correspond to intervals in semitones.

chord cannot be VII.

$$D_m[e_m - b_m] \neq \text{VII} \wedge D_m + 1[0] \neq \text{VII} \tag{48}$$

$$\begin{aligned} & D_{m+1}[f_m - b_{m+1} - 1] = \text{V} \\ \wedge\ & S_{m+1}[f_m - b_{m+1} - 1] = \text{Fund. State} \\ \wedge\ & D_{m+1}[f_m - b_{m+1}] = \text{I} \\ \wedge\ & S_{m+1}[f_m - b_{m+1}] = \text{Fund. State} \end{aligned} \tag{49}$$

We link the first and the second progression to the modulation.

$$e_m = f_m - 2 \quad b_{m+1} = s_m \tag{50}$$

Alteration modulation An alteration modulation introduces a note from the second tonality that is not present in the original tonality to start the modulation. This chord has to be followed by the V of the new tonality, affirming it (see Figure 2c). If the chord used to introduce the alteration cannot be followed by V, it has to be the next chord. The last chord of the first progression must be diatonic, cannot be VII and cannot have a seventh. The first chord of the new progression must be diatonic, and cannot be V or VII.

$$\begin{aligned} D_m[e_m] \neq \text{VII} \ \wedge\ H_m[e_m] = 0 \\ \wedge\ D_{m+1}[0] \notin \{\text{V,VII}\} \end{aligned} \tag{51}$$

$$D_{m+1}[0] < \text{VII} \wedge D_{m+1}[0] \neq \text{V} \tag{52}$$

Where $H_m[c_m]$ is true when chord c_m has a seventh.

We must also ensure that the first chord of the new progression contains a note that is not in the first tonality. We define a function $f_t(n)$ that takes as argument a note in [C,B], and returns the quality of the diatonic chord built on that note if it is in *t*. The function is not defined if the note is not in *t*.

$$f_t(n) = \begin{cases} n \in t & Q_t(n) \\ n \notin t & \perp \end{cases}$$

This is equivalent to a 12-value array, containing for each note the quality of the chord based on this note in t if it exists, and nothing otherwise. We then impose the constraint:

$$f_{t_m}(R_{m+1}[0]) = \perp \ \vee f_{t_{m+1}}(R_{m+1}[0]) \neq f_{t_m}(R_{m+1}[0]) \tag{53}$$

which means that the quality of the chord based on note $R_{m+1}[0]$ cannot be the same in both tonalities. If this note is not in t_m, this is trivially satisfied. This ensures that there is at least one note in the first chord in the new progression that is not in the previous tonality. We still have to enforce that this altered chord is followed by V. Depending on which degree it corresponds to, it might not be possible for V to follow directly. In that case, it should be the next chord.

$$T[D_{m+1}[0], \text{V}] = 0 \implies D_{m+1}[2] = \text{V} \tag{54}$$

$$T[D_{m+1}[0], \text{V}] \neq 0 \implies D_{m+1}[1] = \text{V} \tag{55}$$

Since the modulation possibly affects three chords (the one that introduces the alteration, and the next two), the length of this modulation is considered to be three even if sometimes the modulation is over after the second chord. We link the first and the second progression to the modulation.

$$e_m = s_m - 1 \quad b_{m+1} = s_m \tag{56}$$

Chromatic modulation This kind of modulation occurs when one chord in the first tonality is followed by the V of the new tonality, with a chromatic movement in the voice that plays the leading tone of the new tonality in the dominant chord (see Figure 2d). The voice leading aspect of this modulation needs to be handled in the third (voicing) layer of the Harmoniser project. Similarly to the preparation of diatonic seventh chords, constraints still need to be enforced in this model to make sure that this chromatic movement is possible. In particular, we must enforce that the first chord of the new progression is V, and there must be a one chord overlap between the progressions to ensure that the transition is smooth. The chord in this overlap is thus a secondary dominant in the first tonality, and the dominant in the new one. We must also ensure that the note in the first tonality corresponding to the leading tone in the new tonality is present in the chord just before the dominant of the new tonality (i.e., when modulating from C major to A major, there must be a G in the first chord that can move to a G♯ in the second chord). To enforce that, we must compute the interval in semitones between the keys of the two tonalities, and transform that into a degree difference. This is shown in Table 12.

$$\begin{aligned} d = Degs[|t_m.k_m - t_{m+1}.k_{m+1}|] \\ s = 6 + d \mod 7 \\ D_{m+1}[0] = V \ \wedge\ Ro_m[n_m - 2] = s \\ \vee\ Ti_m[n_m - 2] = s \vee Fi_m[n_m - 2] = s \end{aligned} \tag{57}$$

Where s is the degree that the seventh of the new tonality corresponds to in the first tonality.

We link the first and the second progression to the modulation.

$$e_m = s_m + 1 \quad b_{m+1} = s_m + 1 \tag{58}$$