An Argument based Creative Assistant for Harmonic Blending

Citation for published version:

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Publisher's PDF, also known as Version of record

Published In:
Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
An Argument-based Creative Assistant for Harmonic Blending
Maximos Kaliakatsos-Papakostas\textsuperscript{a}, Roberto Confalonieri\textsuperscript{b}, Joseph Corneli\textsuperscript{c}, Asterios Zacharakis\textsuperscript{a} and Emilios Cambouropoulos\textsuperscript{c}

\textsuperscript{a}Department of Music, Aristotle University of Thessaloniki, Greece
\textsuperscript{b}Artificial Intelligence Research Institute, I3A-CSIC, Bellaterra (Barcelona), Spain
\textsuperscript{c}Department of Computing, Goldsmiths, University of London, UK
\{max, aszachar, emilios\}@mus.auth.gr
confalonieri@iiia.csic.es
j.corneli@gold.ac.uk

Abstract
Conceptual blending is a powerful tool for computational creativity where, for example, the properties of two harmonic spaces may be combined in a consistent manner to produce a novel harmonic space. However, deciding about the importance of property features in the input spaces and evaluating the results of conceptual blending is a nontrivial task. In the specific case of musical harmony, defining the salient features of chord transitions and evaluating invented harmonic spaces requires deep musicological background knowledge. In this paper, we propose a creative tool that helps musicologists to evaluate and to enhance harmonic innovation. This tool allows a music expert to specify arguments over given transition properties. These arguments are then considered by the system when defining combinations of features in an idiom-blending process. A music expert can assess whether the new harmonic idiom makes musicological sense and re-adjust the arguments (selection of features) to explore alternative blends that can potentially produce better harmonic spaces. We conclude with a discussion of future work that would further automate the harmonisation process.

Introduction
The invention of new harmonic spaces in this paper is conceived as a computational creative process according to which a new harmonic idiom is created by means of blending the ‘atoms’ of harmony, i.e., transitions between chords. The blended transitions are created by combining the features characterising pairs of transitions belonging to two idioms (expressed as sets of potentially learned transitions) according to an amalgam-based algorithm (Confalonieri et al., 2015a; Eppe et al., 2015b) that implements Fauconnier and Turner (2002)’s theory of conceptual blending. The transitions are then used in an extended harmonic space that accommodates the two initial harmonic spaces, linked with the new blended transitions.

When modeling creative processes computationally, one of the key questions is how good are the created artefacts. The approach to evaluation that has been applied most frequently within computational creativity requires a human to evaluate attributes of the created work or the system’s operation. Basic measures consider the typicality of a generated artefact within a particular genre, or the quality of the generated work according to the users’ aesthetic judgement (Ritchie, 2007).

In music blending, the evaluation of artefacts is not a trivial matter. This is due not only to the time evolving nature of the final output, but also to the lack of clearly defined criteria for their assessment. In the particular case of transition\textsuperscript{1} blending, which is how harmonic blending is approached in this paper, the evaluation of the blends is of key importance, in order to produce musically meaningful extended harmonic spaces. To evaluate the set of blended transitions and the corresponding generated extended harmonic space, several musical features need to be taken into account according to indications by musicologists. The importance of each particular feature, however, is not known in advance and musicologists need to make adjustments by experimenting with a large set of test cases.

To ease this task, in this paper, we propose a creative tool (Figure 1) that assists a musicologist with the evaluation of harmonic blends. The system allows a musicologist to specify arguments—abstracting the properties of chords and transitions—and to use them for an iterative evaluation of the blended outcome, based on the transitions that the system proposes in order to connect two (potentially remote) harmonic spaces.

Using arguments to make and explain decisions has been proposed and explored in Artificial Intelligence (Bench-Capon and Dunne, 2007), where an argument is a reason for believing a statement, choosing an option, or doing an action. In most existing works on argumentation, an argument is either considered as an abstract entity whose origin and structure are not defined (Dung, 1995), or it is a logical proof for a statement where the proof is built from a knowledge base (Amgoud and Prade, 2009). The use of argumentation in concept invention is, on the other hand, less frequent. Confalonieri et al. (2015b) use Lakatosian’s reasoning to model dialogues in which users engage to discuss the intended meaning of blended concept.

In our approach, arguments encapsulate desirable properties that the user would like to have in the resulting transition blends. Arguments are specified by the user by answering specific questions over the features of the idioms selected as input for the transition blending process. Providing some

\textsuperscript{1}For the rest of the paper, the term transition will be referring to a pair of chords where one follows the other. For example, G7 → C is a transition describing the perfect cadences in tonal music.
The cognitive theory of conceptual blending by Fauconnier and Turner (2002) has been extensively used in linguistics, music composition (Zbikowski, 2002), music cognition (Antovic, 2009, 2011) and other domains mainly as an analytical tool, which is useful for explaining the cognitive process that humans undergo when engaged in creative acts. According to this theory, human creativity is modeled
As a process by which a new concept is constructed by taking the commonalities among two input spaces into account, to form a so-called generic space, and by projecting their non-common structure in a selective way to a novel blended space, called a blend.

In computational creativity, conceptual blending has been modeled by Goguen (2006) as a generative mechanism, according to which input spaces are modeled as algebraic specifications and a blend is computed as a categorical colimit. A computational framework that extends Goguen’s approach has been developed in the context of the CONcept INVENtion Theory (COINVENT) project (Schorlemmer et al., 2014) based on the notion of amalgams (Ontañón and Plaza, 2010). According to this framework, input spaces are described as sets of features, properties and relations, and an amalgam-based workflow finds the blends (Confalonieri et al., 2015; Eppe et al., 2015). The amalgam-based workflow generalises input concepts until a generic space is found and ‘combines’ generalised versions of the input spaces to create blends that are consistent or satisfy certain properties that relate to the knowledge domain (Figure 2).

Amalgam-based conceptual blending has been applied to invent chord cadences (Eppe et al., 2015; Zacharakis, Kaliakatsos-Papakostas, and Cambouropoulos, 2015). In this setting, cadences are considered as special cases of chord transitions—pairs of chords, where the first chord is followed by the second one—that are described by means of features such as the roots or types of the involved chords, or intervals between voice motions, among others. When blending two transitions, the amalgam-based algorithm first finds a generic space between them (point 1 in Figure 2). For instance, in the case of blending the perfect with the Phrygian cadences (Figure 3)—described by the transitions I1: G7 → C and I2: Bm → C5 respectively—their generic space consists of any transition that has a second chord with the root note C, since this is the root note of both inputs’ second chords (C and C5).

After a generic space is found, the amalgam-based process computes the amalgam of two input spaces by unifying their content. If the resulting amalgam is inconsistent, then it iteratively generalises the properties of the inputs (point 2 in Figure 2), until the resulting unification is consistent (point 3 in Figure 2). For instance, trying to directly unify the transitions I1: G7 → C and I2: Bm → C5 would yield an inconsistent amalgam, since a transition cannot both include and not include a leading note to the second chord’s tonic (which is a property of I1 and the I2 respectively). Therefore, the amalgam-based process generalises the clashing property in one of the inputs (e.g., the property describing the absence of leading note would be left empty in I2) and tries to unify the generalised versions of the inputs again. After a number of generalisation steps are applied (point 2 in Figure 2), the resulting blend is consistent (point 3 in Figure 2). However, it may be the case that the blend is not complete, in the sense that this process may have generated an over-generalised term.

After several blends have been computed, an automated evaluation process ranks them according to some optimality principles (Fauconnier and Turner, 2002, Chapter 16). These principles are a necessary aspect of conceptual blending since they allow to filter interesting blends from the (potentially too) many possible ones.

A complete description of optimality principles is out of the scope of this paper and the reader is referred to Goguen and Harrell (2010) for applications of such principles in the Alloy algorithm. We give, however, two extreme examples of ‘bad blends’ for clarifying the importance of using optimality principles in conceptual blending. Example 1: Each of the input spaces is a trivial form of a (bad) blend, since no information from the other input spaces is considered. Example 2: A blend that includes all properties of the generic space, but no other information of the inputs spaces is a bad blend, since it has the least possible connections with the input spaces. These examples suggest two criteria for ranking the blends; we provide a computational characterisation of them below.

\(^3\)http://www.coinvent-project.eu

\(^4\)In the process of blending through amalgams, the notions of ‘amalgam’ and ‘blend’ are the same. Therefore, in the following paragraphs they are used interchangeably.

\(^5\)The amalgam-based algorithm produces many blends by following alternative generalisation paths.
Chord transitions description and blending

Individual chord transitions are the ‘atoms’ of the methodology followed herein to construct new transition matrices. Specifically, transition sets from two musical idioms provide input transitions for blending, producing a list of blended transitions that are afterwards embedded in an extended harmonic space. This methodology is described briefly in the next section while some definitions regarding chord transitions follow.

**Definition 1.** A chord transition $c$ is described by a set of features $\mathcal{F}$.

In this work a transition is represented by 17 features. Features 1-6 refer to the involved chords. Features 8 to 10 indicate changes during the transitions and are based on the Directed Interval Class (DIC) vector (Cambouropoulos, Katsivalos, and Tsougras, 2013; Cambouropoulos, 2012). Feature 7 accounts for the change that occurred regarding the chords’ root notes. The features considered important in this work are the following:

1. $\text{fromRoot}$: the root pitch class of the first chord,
2. $\text{toRoot}$: the root pitch class of the second chord,
3. $\text{fromType}$: the type of the first chord (GCT base),
4. $\text{toType}$: the type of the second chord (GCT base),
5. $\text{fromPCs}$: the pitch classes included in the first chord,
6. $\text{toPCs}$: the pitch classes included in the second chord,
7. $\text{DICinfo}$: the DIC vector of the transition,
8. $\text{DICchas}$: Boolean value indicating whether the DIC of the transition has 0,
9. $\text{DIChas1}$: As above but for DIC value 1,
10. $\text{DIChasMinus1}$: As above but for DIC value $-1$,
11. $\text{ascSemZero}$: Boolean value indicating whether the first chord has the relative pitch class value 11,
12. $\text{descSemZero}$: As above but value 1,
13. $\text{semZero}$: As above but for value 11 or 1,
14. $\text{ascSemNextRoot}$: Boolean value indicating whether the first chord has a pitch class with ascending semitone relation with the pitch class of the second chord’s root,
15. $\text{descSemNextRoot}$: As above but with descending semitone,
16. $\text{seNextRoot}$: As above but with either ascending or descending semitone, and
17. $\text{5thRootRelation}$: Boolean value indicating whether the first chord’s root note is a fifth above of the second’s.

Each feature can be considered as a function that assigns a value to a chord transition $c$. Features’ values are defined differently depending on the properties they represent. For instance, features 3 to 8 are set-value functions that assign a set of values to a chord. We refer to them as $F_i(c)$. The value of the feature 7 is a vector and we refer to it as $\vec{f}(c)$.

Finally, all the other features are binary functions and we refer to them as $f_i(c)$.

From transition blends to transition matrices

In the literature, an effective and common way to describe chord progressions in a music idiom in a statistical manner is by using first-order Markov models (see Kaliakatsos-Papakostas and Cambouropoulos (2014); Simon, Morris, and Basu (2008), among others). Such models reflect the probabilities of each chord following other chords in the idiom, as trained or statistically measured throughout all the pieces in the examined idiom. In this context, individual transitions play an important role on indicating particular characteristics of an idiom.

A convenient way to represent a first order Markov model is through transition matrices, which include one respective row and column for each chord in the examined idiom. The probability value in the $i$-th row and the $j$-th column exhibits the probability of the $i$-th chord going to the $j$-th —the probabilities of each row sum to unit. The utilised chords are actually represented by chord group exemplars, obtained by the method described in Kaliakatsos-Papakostas et al. (2015), while transitions between chords that pertain to the same chord group are disregarded. The representation of chords is based on the General Chord Type representation (Cambouropoulos, Kaliakatsos-Papakostas, and Tsougras, 2014).

Then, an important question is: **How would a blended idiom be expressed in terms of a transition matrix, provided that the transition matrices of two initial idioms are available?**

The idea examined in the present system is to create an extended transition matrix that includes new transitions that allow moving across chords of the initial idioms by potentially using new chords. The examined methodology uses transition blending to create new transitions that: (a) maximally preserve the common parts of transitions between the two initial spaces, and (b) incorporate blended characteristics for creating a smooth ‘morphing’ harmonic effect when moving from chords of one space to chords of the other. An abstract illustration of an extended matrix is given in Figure 4.

By analysing the graphical representation of an extended matrix as depicted in Figure 4 the following facts are highlighted:

1. By using transitions in $\mathcal{I}_i$, only chords of the $i$-th idiom are used. When using these transitions, the resulting harmonisations preserve the character of idiom $i$.
2. Transitions in $\mathcal{A}_{i-j}$ create direct jumps from chords of the $i$-th to chords of the $j$-th idiom. Blended transitions in $\mathcal{A}_{i-j}$ can be directly included in the extended matrix.
3. Transitions in $\mathcal{B}_{i-x}$ constitute harmonic motions from a chord of idiom $i$ to a newly created chord by blending. Similarly, transitions in the $\mathcal{B}_{x-j}$ arrive at chords in idiom $j$ from new chords. For moving from idiom $i$ to idiom $j$ using one external chord $c_x$ that was produced by blending, a chain of two transitions is needed: $c_i \rightarrow c_x \rightarrow c_j$, where $c_i$ in idiom $i$ and $c_j$ in idiom $j$ respectively. A chain of two consecutive transitions with one intermediate external chord from chords of $i$ to chords of $j$ will be denoted as $\mathcal{B}_{i-x-j}$.
4. Sector C transitions are disregarded since they incorporate pairs of chords that exist outside the i-th and j-th idioms, violating our hypothesis for moving from one known chord sets to the other using one new chord at most.

Based on this analysis of the extended matrix, a methodology is proposed for using blends between transitions in I₁ and I₂. Thereby, transitions in I₂ are blended with those in I₁ and a certain number of best blends is stored for further investigation, creating a pool of best blends. Based on multiple simulations, a large number of the best blends (i.e. 100) in each blending simulation should be inserted in the pool of best blends (B), so that several commuting scenarios can be created between the initial transition spaces. Thus, a greater number of blends in the pool of best blends introduces a larger number of possible commuting paths in A₁−j or in B₁−x−j.

**Evaluation of transition blending via arguments**

By applying the aforementioned blending process a pool of best blends is created that is afterwards used for connecting the transition blocks of two initial idioms, through forming an extended matrix. When a music expert is using the system, she/he is able to select pairs of initial input idioms⁶, choose which aspects of blending are important through arguments (analysed in the following paragraphs) and evaluate/re-adjust this choice by observing the produced results in the extended matrix.

The user evaluates the importance of several transition features by answering questions based on the connecting transitions produced by blending in the extended matrix. The features related to the transitions and their constituent chords are classified into 9 questions (Table 1).

**Table 1: Abstraction of chords’ and transition changes’ features.**

<table>
<thead>
<tr>
<th>Question</th>
<th>Chord Properties</th>
<th>Transition Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td></td>
<td>fromRoot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>toRoot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fromType</td>
</tr>
<tr>
<td></td>
<td></td>
<td>toType</td>
</tr>
<tr>
<td>Q2</td>
<td></td>
<td>fromRelPCs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>toRelPCs</td>
</tr>
<tr>
<td>Q3</td>
<td></td>
<td>DIChas0</td>
</tr>
<tr>
<td>Q4</td>
<td></td>
<td>DIChas1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DIChasN1</td>
</tr>
<tr>
<td>Q5</td>
<td></td>
<td>DIChas2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DIChasN2</td>
</tr>
<tr>
<td>Q6</td>
<td>DICinfo</td>
<td></td>
</tr>
<tr>
<td>Q7</td>
<td>ascSemZero</td>
<td></td>
</tr>
<tr>
<td></td>
<td>descSemZero</td>
<td></td>
</tr>
<tr>
<td></td>
<td>semZero</td>
<td></td>
</tr>
<tr>
<td>Q8</td>
<td>ascSemNextRoot</td>
<td></td>
</tr>
<tr>
<td></td>
<td>descSemNextRoot</td>
<td></td>
</tr>
<tr>
<td></td>
<td>semNextRoot</td>
<td></td>
</tr>
<tr>
<td>Q9</td>
<td>5thRootRel</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Question</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>Are roots and types of chords important?</td>
<td></td>
</tr>
<tr>
<td>Q2</td>
<td>Are individual pitch classes of chords important?</td>
<td></td>
</tr>
<tr>
<td>Q3</td>
<td>Are repeating pitch classes in transitions important?</td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td>Are semitone steps in transitions important?</td>
<td></td>
</tr>
<tr>
<td>Q5</td>
<td>Are tone steps in transitions important?</td>
<td></td>
</tr>
<tr>
<td>Q6</td>
<td>Are the intervalic contents of transitions important?</td>
<td></td>
</tr>
<tr>
<td>Q7</td>
<td>Are semitone motions to the tonic important?</td>
<td></td>
</tr>
<tr>
<td>Q8</td>
<td>Are semitones to the second chord’s root important?</td>
<td></td>
</tr>
<tr>
<td>Q9</td>
<td>Are motions of the chord roots by 5th important?</td>
<td></td>
</tr>
</tbody>
</table>

The first two questions concern characteristics of the chords that constitute the transition (features 1 to 6), while the remaining seven questions concern intervalic changes that occur within the transition (features 7 to 17). Relating questions to transition features was performed with the involvement of music experts, to ensure that the mapping is as accurate and as informative to the user as possible.

We denote the set of questions available to the user as Q. When a user selects a question, an argument is automatically generated. For the sake of this paper, an argument is defined as follows.

**Definition 2.** An argument A is a tuple ⟨q, F⟩, where q ∈ Q and F ⊂ F.

The user can specify at most 9 arguments, each of them is mapped to a set of properties. The set of user arguments \{A₁, ..., A₉\} corresponding to answers to Q will be denoted by A. We assume to have a function ψ : A → F that returns the set of chord and transition properties associated with an argument (e.g., for the purposes of the current analysis, Table 1 specifies ψ as a look-up function). The arguments are used to compute two criteria in order to rate a blend: total association and symmetry.
The total association indicates the total number of properties that a blend inherits from the inputs. A blend with higher input associations is preferable since it is structurally more deeply related with the inputs. The total association is calculated by taking the individual association of a blend w.r.t. the input chord transitions into account. The individual association of a blend \( b \) w.r.t. to an input \( I \), denoted as \( a(b, I) \), is defined as:

\[
a(b, I) = \sum_{A_i \in A} \text{Val}(A_i, b, I)
\]

where \( \text{Val} : A \rightarrow \mathbb{R} \) is a function that takes an argument as input and aggregates the values of the chord and transition change properties associated with the argument, by interpreting them according to some music background knowledge. Depending on the type of argument, \( \text{Val} \) is defined in different ways.

When an argument refers to the roots and types of chords \((A_1)\), \( \text{Val} \) is defined as:

\[
\text{Val}(A_1, b, I) = \sum_{F_j \in \psi(A_1)} \text{equals}(F_j(I), F_j(b))
\]

The value of \( A_1 \) is calculated by counting how many properties — among fromRoot, toRoot, fromType and toType — are equals between a blend \( b \) and an input \( I \). equals is a function that returns 1 when two sets are equals and 0 otherwise.

When an argument refers to the individual pitch classes of chords \((A_2)\), \( \text{Val} \) is defined as:

\[
\text{Val}(A_2, b, I) = \sum_{F_j \in \psi(A_2)} |F_j(I) \cap F_j(b)|
\]

The value of \( A_2 \) is calculated as the number of elements that are common in the set-value properties fromRelPCs and toRelPCs of a blend \( b \) and an input \( I \).

When an argument refers to the intervallic contents of transitions \((A_6)\), \( \text{Val} \) is defined as:

\[
\text{Val}(A_6, b, I) = \text{norm}_{[0,1]}(\rho_{f(I),f(b)})
\]

The value of \( A_6 \) is calculated as the Pearson’s correlation coefficient of the vector-value property DICInfo of a blend \( b \) and an input \( I \). Higher correlations in the DIC vectors of two transitions indicate higher resemblance; norm is a function that normalises the Pearson’s coefficient from the interval \([-1, 1]\) to the interval \([0, 1]\). For all the other types of arguments, \( \text{Val} \) is defined as:

\[
\text{Val}(A_i, b, I) = \sum_{F_j \in \psi(A_i)} 1 - (f_j(I) - f_j(b))
\]

Based on the above definitions, the total association value is the sum of the individual associations.

\[
\text{assoc}(b) = \sum_{I \in \mathcal{I}} a(b, I_i)
\]

where \( \mathcal{I} \) is the set of input spaces, containing in this specific case, \( I_1 \) and \( I_2 \).

Symmetry, on the other hand, reflects the balance of properties that a blend inherits from both input spaces. A blend has a high symmetry when it inherits an almost equal proportion of properties from both input spaces. Blends having higher symmetry are preferred to those with lower symmetry, since a high symmetry reflects a stronger hybridisation of structural characteristics. Hybridisation is an important principle to evaluate transition blends.

The blend symmetry is defined in terms of its ‘asymmetry’. The asymmetry of a blend w.r.t. the inputs, denoted as asym\((b)\), is calculated as:

\[
\text{asym}(b) = \frac{a(b, I_1)^2 + a(b, I_2)^2 - 2a(b, I_1)a(b, I_2)}{a(b, I_1)^2 + a(b, I_2)^2}
\]

The value of asym\((b)\) is defined in \([0, 1]\), where 0 stands for a perfect symmetry (equal association with both inputs) and 1 stands for total asymmetry (association only with one input). Additionally, the non-absolute version of the above equation suggests the prevailing input, with a negative value indicating dominating association of the blend with the first input and a positive value contrarily.

The total rate of a blend is computed by taking the input association and asymmetry values into account.

\[
\text{rate}(b) = \frac{\text{assoc}(b)(1 - \text{asym}(b))}{\text{assoc}(b) + (1 - \text{asym}(b))}
\]

The above expression promotes pairs of association and symmetry that are both high, while a simple sum would allow a low value of the one to be covered by the other.

Finally, a decision making criterion to compare any pair of blends \( b_1, b_2 \in \mathcal{B} \) can be defined as follows.

**Definition 3 (Decision criterion).** A blend \( b_1 \) is preferred to a blend \( b_2 \) if and only if \( \text{rate}(b_1) \geq \text{rate}(b_2) \).

It is worthy to notice that the above criterion guarantees that any pair of blends is comparable, and, consequently, it allows to decide which blends are the best ones. This is an important property for blend evaluation and, generally, for approaches to argumentation-based decision making (Amgoud and Prade, 2009; Bonet and Geffner, 1996).

**System evaluation**

Referring to Figure 1, via the interface \( \alpha \), the user has access to modules \( \gamma \) and \( \beta \) which can be used to specify concepts that will inform the resulting product, namely, the input idioms and arguments that impose constraints on the generated blend. These are translated by the system into process-friendly formats. Module \( \epsilon \) embodies the (process-level) concept of a system that make use of the supplied idioms and the blending properties to generate example transition matrices, \( \mathcal{C} \). In the current version of the system, these transitions are evaluated by the user (music expert) in step \( \epsilon \) using sophisticated harmonic knowledge that reflects an historically established musical aesthetic. The user can then return to the GUI \( \alpha \), and adjust the settings of \( \gamma \) and \( \beta \) to regenerate the transitions.

This is illustrated in Figure 5 in box \textbf{P1}, using the diagrammatic extension to the FACE model by Colton et al. (2014). Here, capital letters \( F, A, C, \) or \( E \) are creative
acts that generate a framing, aesthetic, concept, or example, respectively. Administrative acts $S$ and $T$ denote selection and translation. Lower-case letters denote the generated artefact in each case (e.g., the concept $c$ corresponding to the concept-creation act $C$). Subscripts $p$, $g$, or $m$ indicate whether the act takes place at the process, ground, or meta level. Inside each box, stacks show the dependence in development epochs, and arrows show run-time message passing. Acts taken by the programmer or user are decorated with a bar, whereas acts taken by the system itself receive no extra decoration.

In the current version of the system, apart from the programmer’s creative acts specifying the modules and their interconnections, and the algorithm $C_p^g$ that turns inputs into blends, the user, who is assumed to be a music expert, must intervene in the system in two places.

First, the user defines system settings $C_g^p$, $C_g^g$ that correspond to the selection of input idioms and of arguments respectively. Second, after the run completes, he or she evaluates the system output via $A_g^g$.

The system’s primary responsibilities take place through the creative acts $E_g^g$, which generate blends, and $S[a_{g}^{β,γ}](e^*)$, in which the aesthetic $A_{g}^{β,γ}$ (a unified label for assoc and asym, which are defined anew in each run, based on a fixed translation of the user’s arguments, as specified in the previous section) is applied to rate the possible blends, and select to a final extended transition matrix.

Therefore, the key idea behind what has been implemented so far is an ‘automated ranking/evaluation’ step that guides the selection of blends, $S[a_{g}^{β,γ}](e^*)$ according to the arguments defined by the user. The development of the programmatic components that operationalise this process has relied on both computer science and musicological insights. This approach has been characterised as meaningful per se through informal feedback provided by musical experts – but is perhaps especially valuable because it constitutes a prototype for more involved automated evaluation of computer-generated harmonic spaces.

Indeed, the next step towards the development of a more autonomously creative system using the same architecture is fairly clear: future work would need to ‘close the loop’ computationally, connecting the evaluation of generated transition matrices with the parameter-setting (i.e., argumentation) stage, and making this run autonomously to refine the system’s behaviour. This as-yet hypothetical situation is illustrated in the box **P2**.

Here, the programmer has translated some of the user-specified aesthetics into code $T[A_g^g]$, and invented a meta-level concept $C_g^g$, defining a system component that can automatically apply these aesthetics to the generated transition matrices $e_g^g$ as in order to automatically generate new system settings $C_g^g$, $C_g^g$.

**Conclusion, Discussion and Future Work**

In this paper, we described a methodology for harmonic blending and we proposed a creative system that assists musicologists with the evaluation and enhancement of harmonic innovation. We defined some harmonic features of chord transitions utilised for evaluating blends of transitions, leading to the invention of novel harmonic spaces. The system allows a musicologist to specify arguments over these features that are taken into account in the generation of new harmonic spaces. The music expert can then assess whether the new harmonic idiom makes musicological sense and re-adjust the arguments to explore alternative blends that can potentially produce better harmonic spaces.

The main advantage of the current system is the agile interaction through which the user can express desirable properties over the transition blends and their argument-based evaluation in order to produce musically meaningful results. The added value of argumentation is the ranking/evaluation of blended transition—obtained by conceptual blending of two input transition belonging to two musical idioms—by answering questions which abstract several properties of chord transitions. On the other hand, the evaluation of the creative output of the system, i.e., an extended harmonic space that includes blended transitions, is carried out by the user via an introspective argumentative dialogue.

In a future work we intent to use the argumentation-based process for evaluating the blended harmonisations of user defined melodies, i.e., actual music output. Additionally, mapping the properties of the blended idiom or, at a latter stage of a harmonised melody, back to the parameter-setting stage opens an interesting direction for future research and further improvements of the system. The added value of argumentation can be stressed, for instance, by letting the system suggest possible refinements of the initial user arguments, progressively converting part of the introspective user evaluation into a more explicit format. For example, a future version of the system would be based on identifying harmonic features of the input spaces that automatically suggest an ‘optimal’ set of initial arguments. The current version of the system is an already-usable prototype on the way towards the development of a more autonomous creative system.

**Acknowledgments**

This work is partially supported by the COINVENT project (FET-Open grant number: 611553).

**References**


