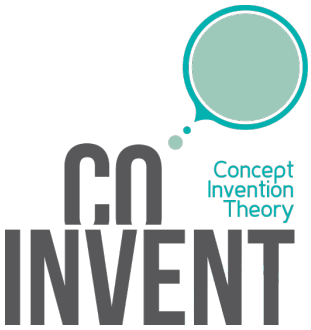


D2.2
Concept Discovery in Rich Backgrounds: Values, Argumentation, and Coherence in Evaluating Concept Invention

Editors	Roberto Confalonieri, Enric Plaza, Marco Schorlemmer
Reviewers	Joseph Corneli

Grant agreement no.	611553
Project acronym	COINVENT Concept Invention Theory
Date	June 22, 2016
Distribution	PU

Disclaimer

The project COINVENT acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open Grant number 611553.

Abstract

This deliverable describes the efforts and work achieved for the Tasks 2.2, 2.4 and 2.4 within the COINVENT project. The different articles attached tackle the process of concept invention from different perspectives using arguments, values, audiences and conceptual coherence.
Keyword list: **Rich Background**, **discovery**, **arguments**, **coherence**

Executive Summary

In this deliverable we tackle two open questions in concept invention processes, in which we use conceptual blending as a tool for modeling the creation of new concepts. On the one hand, we investigate how a Rich Background can support the discovery of concepts to be blended. On the other hand, we study how computational argumentation and coherence can be used to evaluate creative artefacts.

The deliverable consists of the following articles, three of them published in the International Conference of Computational Creativity (editions 2015 and 2016) and one of them submitted to the C3GI Computational Creativity, Concept Invention, and General Intelligence Workshop, 2016:

1. R. Confalonieri, J. Corneli, A. Pease, E. Plaza, M, Schorlemmer. *Using Argumentation to Evaluate Concept Blends in Combinatorial Creativity.* In Proceedings of the 6th International Conference on Computational Creativity, ICCC 2015.

2. R. Confalonieri, E. Plaza, M, Schorlemmer. *A Process Model for Concept Invention.* Accepted in the 7th International Conference on Computational Creativity, ICCC 2016.

3. M, Schorlemmer, R. Confalonieri, E. Plaza, *Coherent Concept Invention.* Submitted to the C3GI Computational Creativity, Concept Invention, and General Intelligence Workshop, 2016.

4. M. Kaliakatsos Papakostas, R. Confalonieri, J. Corneli, A.Zacharakis, E.Cambouropoulos. *An Argument-based Creative Assistant for Harmonic Blending.* Accepted in the 7th International Conference on Computational Creativity, ICCC 2016.

In [1], we motivate the use of computational argumentation for evaluating concept blends and other forms of combinatorial creativity. We exemplify our approach in the domain of computer icon design, where icons are understood as creative artefacts generated through concept blending. We present a semiotic system for representing icons, showing how they can be described in terms of interpretations and how they are related by sign patterns. The interpretation of a sign pattern conveys an intended meaning for an icon. This intended meaning is subjective, and depends on the way concept blending for creating the icon is realised. We show how the intended meaning of icons can be discussed in an explicit and social argumentation process modeled as a dialogue game, and show examples of these following the style of Lakatos. In this way, we are able to evaluate concept blends through an open-ended and dynamic discussion in which concept blends can be improved and the reasons behind a specific evaluation are made explicit. In the closing section, we explore argumentation and the potential roles that can play at different stages of the concept blending process.

In [2], we propose a computational framework that models concept invention. The framework is based on conceptual blending, a cognitive theory that models human creativity and explains how new concepts are created. Apart from the blending mechanism modeling the creation of new concepts, the framework considers two extra dimensions such as origin and destination. For the former, we describe how a Rich Background supports the discovery of input concepts to be blended. For the latter, we show how arguments, promoting or demoting the values of an audience, to which the invention is headed, can be used to evaluate the candidate blends created. Throughout the paper, we exemplify the computational framework in the domain of computer icons.

In [3], we address the problem on how newly invented concepts are evaluated with respect to a background ontology of conceptual knowledge so as to decide which of them are to be accepted into a system of familiar concepts, and how this, in turn, may affect the previously accepted conceptualisation. As technique to tackle this problem we explore the applicability of Paul Thagard's computational theory of coherence. In particular, we propose a formalisation of Thagard's notion of conceptual coherence for concepts represented in the $\mathcal{AL}$ description logic and explore by means of an illustrative example the role coherence may play in the process of conceptual blending.

In [4], we describe a tool that assists music experts in the evaluation of harmonic blending by means of arguments. 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.

Using Argumentation to Evaluate Concept Blends in Combinatorial Creativity

Roberto Confalonieri[1], Joseph Corneli[2], Alison Pease[3], Enric Plaza[1] and Marco Schorlemmer[1]

[1]Artificial Intelligence Research Institute, IIIA-CSIC, Spain
[2]Computational Creativity Research Group, Department of Computing, Goldsmiths, University of London, UK
[3]Centre for Argument Technology, School of Computing, University of Dundee, UK

Abstract

This paper motivates the use of computational argumentation for evaluating 'concept blends' and other forms of combinatorial creativity. We exemplify our approach in the domain of computer icon design, where icons are understood as creative artefacts generated through concept blending. We present a semiotic system for representing icons, showing how they can be described in terms of interpretations and how they are related by sign patterns. The interpretation of a sign pattern conveys an intended meaning for an icon. This intended meaning is subjective, and depends on the way concept blending for creating the icon is realised. We show how the intended meaning of icons can be discussed in an explicit and social argumentation process modeled as a dialogue game, and show examples of these following the style of Lakatos (1976). In this way, we are able to evaluate concept blends through an open-ended and dynamic discussion in which concept blends can be improved and the reasons behind a specific evaluation are made explicit. In the closing section, we explore argumentation and the potential roles that can play at different stages of the concept blending process.

Introduction

A proposal by (Fauconnier and Turner, 1998) called *concept blending* has reinvigorated studies trying to unravel the general cognitive principles operating during creative thought. According to (Fauconnier and Turner, 1998), concept blending is a cognitive process that serves a variety of cognitive purposes, including creativity. In this way of thinking, human creativity can be modeled as a blending process that takes different mental spaces as input and blends them into a new mental space called a *blend*. This is a form of *combinatorial creativity*, one of the three forms of creativity identified by Boden (2003). A blend is constructed by taking the existing commonalities among the input mental spaces (called the generic space) into account, and by projecting the structure of the input spaces in a selective way. In general the outcome can have an emergent structure arising from a non-trivial combination of the projected parts. Different projections lead to different blends and different generic spaces constrain the possible projections.

This poses challenges from a computational perspective: large number of possible combinations exhibiting vastly different properties can be constructed by choosing different input spaces, using different ways to compute the generic space, and selecting projections. Within the Concept Invention Theory project[1] (COINVENT), we are currently developing a computational account of concept blending based on insights from psychology, Artificial Intelligence (AI), and cognitive modelling (Schorlemmer et al., 2014). One of our goals is to address this combinatorial nature. One potential outcome of this work is a deeper understanding of the way combinatorial creativity works in general.

The formal and computational model for concept blending under development in COINVENT (Bou et al., 2014) is closely related to the notion of *amalgam* (Ontañón and Plaza, 2010). Amalgamation has its root in case-based reasoning and focuses on the issue of combining solutions coming from multiple cases. Assuming the solution space can be characterised as a generalisation space, the amalgam operation combines input solutions into a new solution that contains as much information from the two inputs solutions as possible. When input solutions cannot be combined, amalgamation generalises them by dropping some of their properties. This process of generalisation and combination can be expensive from a computational point of view, depending on the search space to be explored.

The amalgam-based approach for computing blends makes explicit the combinatorial nature of concept blending, which raises the issue of evaluating and selecting novel and valuable blends as opposed to those combinations that lack interest or significance. Although Fauconnier and Turner (1998) suggest a number of qualitative criteria that can be used for evaluating concept blends, it is not straightforward to chararacterise them in a computational model.

In this paper, we propose to explore an argumentative approach to understanding and evaluating the meaning, interest, and significance of concept blends. Specifically, we propose to view evaluating blends as a process of *argumentation*, in which the specifics of a blend are pinpointed and opened up as issues of discussion. Our intuition is that in the context of new ideas, proposals, or artworks, people use critical discussion and argumentation to understand, absorb and evaluate. We also consider the constructive roles that argumentation can play in concept blending.

Computational argumentation models have recently appeared in AI applications (Bench-Capon and Dunne, 2007;

[1]See `http://www.coinvent-project.eu` for details.

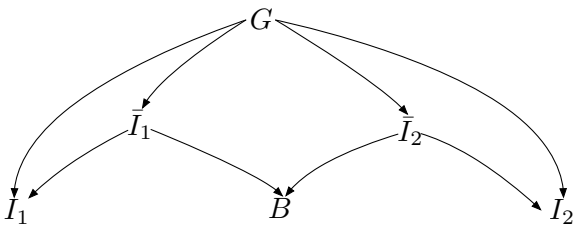


Figure 1: An amalgam diagram with inputs I_1 and I_2 and blend B obtained by combining $\bar{I}_1$ and $\bar{I}_2$. The arrows indicate generalisation.

Rahwan and Simari, 2009), and we believe that incorporating argumentation can foster the development of a fuller computational account of combinatorial creativity. The current paper develops these themes at the level of (meta-) design; implementation is saved for future work.

Roles of Argumentation in Concept Blending

Consider the *amalgam* diagram modeling the concept blending process (Figure 1): two input spaces I_1, I_2, two of their possible generalisations $\bar{I}_1$, $\bar{I}_2$, which have a generic space G and blend B. When two input spaces cannot be combined because they do not satisfy certain criteria, the inputs have to be generalised for omitting some of their specifics. The combination of each specific pair $\bar{I}_1$, $\bar{I}_2$ yields a blend.

Informally, we can imagine argumentation taking place at various points in the amalgam diagram. In general this would happen in response to indeterminacy, that is, when some features of the diagram are underdetermined. We foresee that argumentation can be used:

a. to express opinions or points of view that can be used for guiding the selection/omission of specific parts of the input spaces; in particular, to select a specific pair of generalisation $\bar{I}_1$, $\bar{I}_2$ of the input spaces in the blending process;

b. to provide a computational setting for modeling discussions around the quality of a creative artefact, with the aim of evaluating and refining the generated blends.

In the first case, arguments would be about generalisation, i.e. which features should be preserved from I_1 and which features should be preserved from I_2. More complex inferences could be involved, for example in a case where I_1 is fixed, and constraints and various optimality criteria on the blend are imposed, which then yield various constraints on what the other input I_2 should be. We return to this point in the discussion section, and we focus for the most part on the second case.

In the second case, argumentation would be used to evaluate a range of blends, and the evaluation is carried out *post hoc*, by a variation of try-it-and-see. A range of blends are trialled, each one bringing out different (un)intended meanings. The evaluation is modeled as an argument, or dialogue in which the specifics of a blend are pinpointed and opened up as issues of discussion. This dialogue can be considered as an introspective evaluation, although it usually takes place among several parties as a means for the social development and understanding of creative artefacts. In this paper, we focus on this role.

Our Approach

To exemplify our approach, we take the domain of computer icons into account. We assume that concept blending is the implicit process which governs the creative behavior of icon designers who *create* new icons by blending existing icons and signs. To this end, we propose a simple semiotic system for modeling computer icons. We consider computer icons as combinations of signs (e.g. document, magnifying glass, arrow etc.) that are described in terms of *interpretations*. Interpretations convey *actions-in-the-world* or *concepts* and are associated with shapes. Signs are related by sign-patterns modeled as qualitative spatial relations such as *above*, *behind*, etc. Since sign-patterns are used to combine signs, and each sign can have multiple interpretations, a sign-pattern used to generate a computer icon can convey multiple intended meanings to the icon. These are subjective interpretations of designers when they have to decide what is the best interpretation an icon can have in the real world. In this paper, we show how the intended meaning of new designed (blended) icons can be evaluated and refined by means of Lakatosian reasoning.

Background

Computational argumentation

Computational argumentation in AI aims at modeling the constitutive elements of argumentation, that are i) arguments, ii) attack relations modeling conflicts, and iii) acceptibility semantics for selecting valid arguments (Bench-Capon and Dunne, 2007; Rahwan and Simari, 2009).

The most well-known computational argumentation framework is due to Dung (1995). Dung defines an abstract framework to represent arguments and binary attack relations, modeling conflicts, by means of a graph. He defines different acceptibility semantics to decide which arguments are valid and, consequently, how conflicts can be resolved (Figure 2).

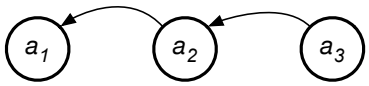


Figure 2: Dung framework example: Nodes represent arguments and edges (binary) attack relations. Argument a_1 is attacked by a_2 which is attacked by a_3. Thus, a_2 is defeated and a_1 can be accepted. a_3 is also accepted.

Abstract argumentation frameworks do not deal with how arguments are generated and exchanged. They merely focus on attack relations between arguments and acceptibility semantics. However, the intrinsic dialectical nature of argumentation is fully explored when an explicit argumentation process is considered. Then, the purpose of a dialogue becomes essential to determine how arguments should be generated and exchanged, and how a dialogue should be structured (Walton and Krabbe, 1995).

Lakatosian argument and dialogue

Lakatos (1976) was a philosopher of mathematics who developed a model of argument, presented as a dialogue, to

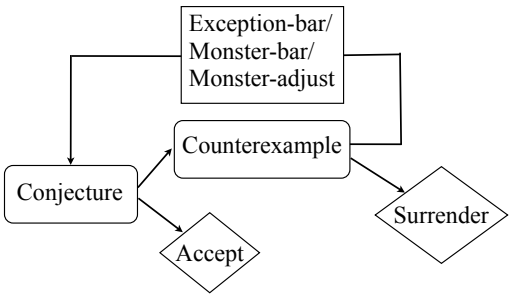


Figure 3: Our interpretation of Lakatos's game patterns.

describe ways in which mathematicians explore and develop new areas of mathematics. In particular, he looked at the role that conflict plays in such explorations, presenting a rational reconstruction of a dialogue in which claims are made and counterexamples are presented and responded to in various different ways. His resulting model describes conceptual continuity and change in the growth of knowledge, and contains dialogue moves, or methods, which suggest ways in which concepts, conjectures and proofs are fluid and open to negotiation, and gradually evolve via an organic process of interaction and argument between mathematicians. These dialogue moves are:

Surrender consists of abandoning a conjecture in the light of a counterexample.

Piecemeal exclusion is an exception-barring method that deals with exceptions by excluding a class of counterexamples, i.e., by generalising from a counterexample to a class of counterexamples which have certain properties.

Strategic withdrawal is an exception-barring method that uses positive examples of a conjecture and generalises from these to a class of object, and then limits the domain of the conjecture to this class.

Monster-barring/monster-adjusting is a way of excluding an unwanted counterexample. This method starts with the argument that a 'counterexample' can be ignored because it is *not* a counterexample, as it is not within the claimed concept definition. Rather, the object is seen as a monster which should not be allowed to disrupt a harmonious conjecture. Using this method, the original conjecture is unchanged, but the meaning of the terms in it may change. Monster-adjusting is similar, in that one reinterprets an object in such a way that it is no longer a counterexample, although in this case the object is still seen as belonging to the domain of the conjecture.

The moves above are not independent processes; much of Lakatos's work stressed the interdependence of creation and justification. These moves describe the evolution of both arguments and conclusions in mathematics, and as such constitute argument patterns, or schemes. However, they are a rational representation of exchanges between mathematicians and describe dynamic, rather than static arguments, presented as a dialogue. Thus, they also have temporal structure, and can be seen as a dialogue game, in which at any point various dialogue moves are applicable (see (Pease et al., 2014) for a description of Lakatos's methods in these terms). The fact that we include negotiations over definitions and changes in the conclusions being argued means that it is difficult to apply traditional abstract argumentation frameworks, which assume that such aspects are stable. However, we can see some of the moves in terms of Dung's framework: for instance if an initial argument for a conjecture forms a_1 in Figure 2, then a_2 might be a counterexample to the conjecture, and a_3 might be the monster-barring move.

The Lakatosian way of conceiving the reasoning as an open-ended discussion about a problem suggests that we can exploit Lakatos's moves for structuring dialogues for the evaluation of creative artefacts. Evaluation in creativity is not a static and rigid process, and the discussion should flow in a dynamic way. As such, in this paper, we propose to use Lakatosian reasoning to model the negotiation about the intended meaning of generated blends (icons). Figure 3 shows the dialogue game we will adopt to model these dialogues. For another formal framework of dialogue games for argumentation see (Prakken, 2005).

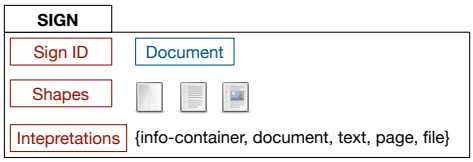
Icons and Signs

We follow a semiotic approach to specify the intended meaning of computer icons. Semiotics is a transdisciplinary approach that studies meaning-making with signs and symbols (Chandler, 2004). Although it is clearly related to linguistics, semiotics also studies other forms of non-linguistic sign systems and how they may convey meaning; this includes not only designation, but also analogy, and metaphor. Although some people may regard Peirce's Sign Theory as the origin of semiotics, Saussure founded his semiotics (semiology) in the social sciences. Currently, cognitive semiotics and computational semiotics take their own perspectives on the relation between sign and meaning-making. In this paper, we take a semiotic approach to describe computer icons in the sense that icons, as a spatial pattern of shapes, are viewed as signs, and compositions of signs are interpreted to convey a meaning, as when we say 'this icon means the download is still active'.

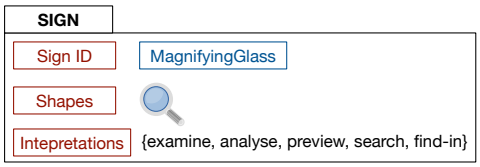
The shapes recurrently used in icons are interpreted as *signs*; screens, magnifying glasses and folders are examples of signs. A magnifying glass sign can be used in different icons in such a way that its meaning is *context-dependent*, that is, it depends on other signs related to it in different icons. We associate to each sign a set of *interpretations*, that encode the kinds of intended meaning associated to that sign as *actions-in-the-world* or *concepts*.

An icon is represented as a pattern defined by a collection of signs and qualitative spatial relations like *above*, *behind*, etc. We can find patterns of meaning that are shared among different icons by analysing recurrent patterns of signs and their spatial relation. We call them *sign patterns*. A sign pattern has an associated collection of *interpretations* that encode the intended meanings associated to that sign pattern.

Signs, sign patterns, and interpretations, which we will use in the paper, can be built by analysing and annotating existing libraries of computer icons. As we shall see, the inherent polysemy of signs, sign patterns and icons opens the way to use arguments for evaluating the quality or adequacy of new icons created by concept blending.

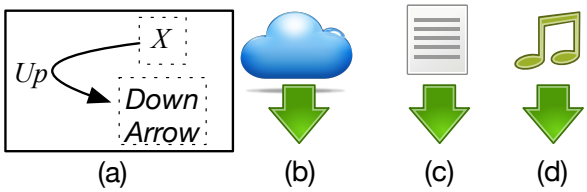


(I) The structure of the DOCUMENT sign, including associated shapes and interpretations.

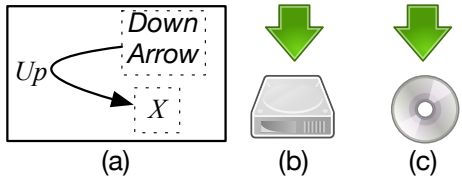


(II) The structure of the MAGNIFYINGGLASS sign, including associated shapes and interpretations.

Figure 4: Example of signs



(I) (a) the sign pattern FROM-DOWNARROW with three examples of the pattern where X is a sign for (b) cloud content, (c) document content, and (d) audio content.



(II) (a) the sign pattern DOWNARROW-TOWARD with two examples of the pattern where X is a sign for (b) a hard disk storage, and (c) an optical disk storage.

Figure 5: Example of different sign patterns used with the same sign DOWNARROW

A semiotic system for icons

In this section, we will formalise the notions presented above. A *sign* S is a tuple $\langle id, \mathcal{F}, \mathcal{A} \rangle$ where id is a sign identifier, $\mathcal{F}$ is a set of shapes embodying the sign S and $\mathcal{A}$ is a set of interpretations. We use $\mathcal{S}$ to denote the available set of signs. Figure 4 provides two examples. Figure 4I shows the structure of the DOCUMENT sign, with several shapes embodying the sign, and a list of interpretations that express how this sign is used in different ways to convey meanings such as *info-container, document, text, page, file*. Intuitively, this means that the shapes used in the icons are sometimes interpreted as a document and other times as a page, etc. Moreover, the specific shapes can be used interchangeably to embody a DOCUMENT, i.e. there is no clear distinction, regarding the shapes, between document vs. page vs. file. Another example of a sign is the MAGNIFYINGGLASS shown in Figure 4II, with interpretations *examine, analyse, preview, search*, and *find-in*.

We will also describe a library of annotated *icons* $\mathcal{I}$, where each icon $I \in \mathcal{I}$ consists of two parts: (1) a spatial configuration of signs and (2) the intended meaning of that icon. For instance, in Figure 5I, the icon (b) has the spatial configuration of a 'cloud on top of a downward-arrow' and its meaning is 'downloading content from the cloud'.

Sign patterns

In our framework, sign patterns relate signs in icons using spatial qualitative relationships such as *above*, *behind*, *up*, *down*, *left*, etc. We assume that these relationships are represented as binary predicates, *Above*(X,Y), *Up*(X,Y), etc., where X and Y are variables ranging over signs in $\mathcal{S}$. For our current purposes, we use the qualitative spatial relationships defined in (Falomir et al., 2012).

Let us consider two examples of sign patterns that include the DOWNARROW sign. DOWNARROW has a vertical downward-pointing arrow shape and is associated with the interpretations {*down*, *downward*, *downloading*, *download-from* and *download-to*}. The sign pattern called FROM-DOWNARROW (shown in the schema labelled (a) in Figure 5I) uses the qualitative spatial relationship *up* between a variable X and the sign DOWNARROW. Examples (b), (c) and (d) in Figure 5I illustrate the intuitive meaning of the sign pattern FROM-DOWNARROW: 'downloading X'. Thus, example (b) refers to downloading cloud content, (c) document content, and (d) audio content.

The inherent asymmetry of arrows in general, and arrow signs particularly, can be appreciated when considering the opposite spatial relation, when the sign DOWNARROW is "up" from another sign (Figure 5II). Then, the sign pattern DOWNARROW-TOWARD is used to mean that X is the destination of the downloading. Example icons (b) and (c) are intended to mean that the data being downloaded (whose type or origin is now elided) is to be stored in a destination such as a hard disk or an optical disk.

Evaluating Blends using Argumentation

As briefly described previously, the amalgam-based computation of concept blending amounts to combine different input spaces into a new space, called blend, by taking the commonalities of the inputs into account, by generalising some of their specifics and by projecting other elements. In the following, we describe how concept blending can account for modeling the creative process of a designer of computer icons.

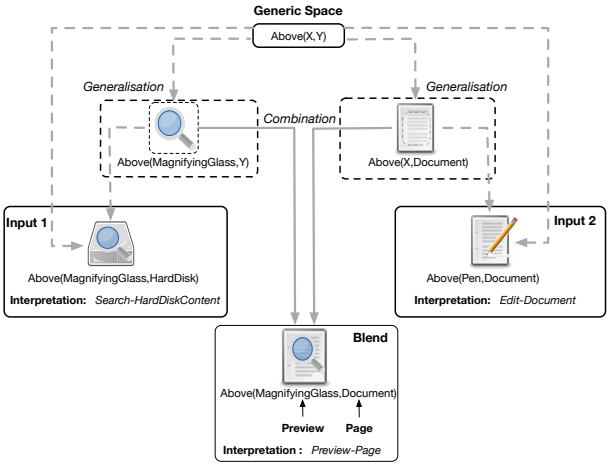


Figure 6: Generating an icon interpreted as *Preview-Page* through amalgam-based concept blending.

A design scenario

Assume a designer is looking for creating a new icon with the intended meaning of previewing a document or a page. The creation of such icon can be achieved by the following amalgam-based concept blending process (Figure 6). In addition to the DOCUMENT and MAGNIFYINGGLASS signs, we assume we have available a HARDDISK sign and a PEN sign which have already been used to make icons.

The input mental spaces. The input mental spaces of the designer are an icon of a hard-disk with a magnifying glass hovering above it, whose meaning is *Search-HardDiskContent*, and an icon of a document with a pen above it, whose meaning is *Edit-Document*.

The generic space. The sign pattern *Above*(X,Y) is used in both icons. The first icon contains the relation *Above*(MAGNIFYINGGLASS, HARDDISK) between the MAGNIFYINGGLASS and the HARDDISK, and the second contains the relation *Above* (PEN,DOCUMENT) between the PEN and the DOCUMENT.

Further generalisation. Two generalisation steps are needed: *Above*(MAGNIFYINGGLASS,HARDDISK) $\rightarrow$ *Above*(MAGNIFYINGGLASS,Y); correspondingly, *Above*(PEN,DOCUMENT) $\rightarrow$ *Above*(X,DOCUMENT).

Combination via variable substitution. We combine the schemas *Above*(MAGNIFYINGGLASS, Y) and *Above*(X, DOCUMENT) via [X/MAGNIFYINGGLASS, Y/DOCUMENT]. The icon of a page with a magnifying glass hovering above it is generated.

The intended meaning. The designer associates to the icon the intended meaning of *Preview-Page*, by selecting the interpretations (*Preview*, *Page*) for the MAGNIFYINGGLASS and DOCUMENT signs.

In this case, the designer decided that the intended meaning of *Above*(MAGNIFYINGGLASS, DOCUMENT) is *Preview-Page*, that is, a page can be examined without opening it. However, during the creative process, the designer could have generated other blends, not only by combining other signs, but also by selecting different interpretations associated to the MAGNIFYINGGLASS and DOCUMENT signs. For instance, the icon in Figure 7 still represents a page with a magnifying glass hovering above it, but it has been given a different intended meaning.

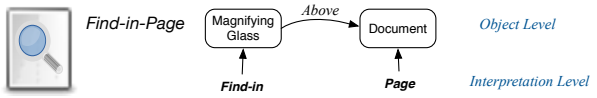


Figure 7: An example of interpreting the sign pattern of an icon as *Find-in-Page*.

The meaning of a blended icon cannot simply be considered right or wrong: interpretation depends on different points of view. Thus the evaluation of whether it is useful or valid for a specific purpose can be the object of a discussion.

Arguments about intended meanings

In the icon domain, arguments may include a clear interpretation of any constituent signs in the icon if it is a composition of signs, or a good fit with other icons in the icon set.

For example, we can consider a counter-argument, i.e. an argument that *attacks* the interpretation a_1 "magnifying glass above document means *Preview-Page*" in Figure 6, to be phrased as follows:

a_2 : "However, the icon in Figure 6 can also be interpreted to mean *Find-in-Page*."

The rationale is that the MAGNIFYINGGLASS sign can often be understood as *finding* or *searching* for something. Thus, the icon can be also interpreted as *Find-in-Page* by associating the interpretation *find-in* instead of *preview* for the same sign MAGNIFYINGGLASS (Figure 7).

This attacking argument can be made at an abstract/conceptual level, for instance, by taking other possible blends of the DOCUMENT and MAGNIFYINGGLASS signs related by the sign pattern *Above*(X,Y) into account. Or, alternatively, if there is an icon library that contains an icon that 'satisfies' the argument above, then this attacking argument can be supported by a specific counterexample. Any of these two forms of attack evaluates negatively the icon in Figure 6. Therefore, if there are several alternative designs for a new icon, this attacking argument diminishes the degree of optimality/adequacy of that design with respect to alternative designs.

The original interpretation can be defended, as usually done in computational argumentation models, by a new argument that attacks the attacking argument a_2. For instance, the designer may say:

a_3 : "The icon in Figure 6 can only be interpreted differently if MAGNIFYINGGLASS is understood to mean *find-in* instead of *preview*. However, the other icons in

my library use MAGNIFYINGGLASS to mean *preview*, not *find-in*."

Argumentation semantics can then be used, once a network of arguments is built, to determine the outcome. For instance whether argument a_1, the original interpretation, is defeated or not can be determined as follows (Figure 2): in this example a_3 has no attack, so it is undefeated, which means it defeats a_2; since a_2 is defeated, the attack against a_1 is invalid and a_1 is undefeated (i.e. is accepted).

Arguments about the intended meanings of an icon can be embedded in a dialogue modeled in terms of Lakatos's moves and the dialogue pattern shown in Figure 3.

Lakatosian reasoning for blend evaluation

Here we present a Lakatos-style dialogue between two players, a proponent P and an opponent O. The goal of each player is to persuade the other player of a point of view, in this paper, the intended meaning of a new blended icon. In such a setting, we expect to see negotiations over the meaning of an icon take place between experts and novices, or between people designing icons and people using (interpreting) them, or various combinations.

To discuss a given icon using Lakatosian reasoning, we assume that an initial conjecture is about the interpretation of an icon usually being an *action-in-the-world* or a *concept*, together with an example of a particular icon and a particular interpretation. The conjecture could be constructed by inductive generalisation.

Example 1. In this example, Lakatosian reasoning is used for discussing the intended meaning of a new icon generated by concept blending:

P_1: "An icon with a magnifying glass over a page means *Preview-Page*" (Conjecture)
O_1: "I disagree, this icon (Figure 7) means *Find-in-page*." (Counterexample)
P_2: "No, this is a different case because the magnifying glass must be over pages with text on them to magnify (it shows what we're about to magnify)." (Monster-barring)

After this dialogue, it is agreed that the intended meaning of the icon is *Preview-Page* and the icon itself has been clarified. Alternatively, the proponent and the opponent could make a different evaluation by following different moves. For instance, if the proponent accepts the counterexample, then the intended meaning of the icon can be refined due to piecemeal exclusion:

P_1: "An icon with a magnifying glass over a page mean *Preview-Page*" (Conjecture)
O_1: "I disagree, this icon (Figure 7) means *Find-in-page*." (Counterexample)
P_3: "Ok, only icons with a magnifying glass over a page with text mean *Preview-Page*". (Piecemeal exclusion)

After this dialogue the intended meaning about the new icon has been changed by modifying the conjecture and taking the counterexample into account.

Sometimes players have different points of view due to the sign patterns they have used in their concept blending.

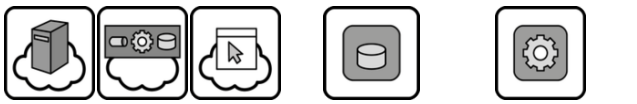


Figure 8: Interpreting the design of cloud icons[2]

Example 2. Let us imagine that the proponent has generated an intended meaning for an icon using the FROM-DOWNARROW sign pattern, whereas the opponent has used the DOWNARROW-TOWARD pattern (Figure 5 illustrates these cases). The two players can engage in the following dialogue:

P_1: "Look at icons in Figure 5I, icons with a DOWNARROW relate to content." (Initial Conjecture)
O_1: "The icon in Figure 5IIb has a DOWNARROW but doesn't relate to content." (Counterexample)
O_2: "The icon in Figure 5IIc also has a DOWNARROW but doesn't relate to content." (Counterexample)
P_2: "The conjecture is right because the two examples actually do relate to content as they are to do with storage and content is part of storage." (Monster-adjusting)

In this case, the proponent excludes the counterexamples using monster-adjusting, and reinterpreting them in a way that they are not counterexamples anymore.

A conjecture might even be at a higher level, for asserting that a particular metaphor is appropriate or inappropriate.

Example 3. For example, someone who is familiar with the 'gear means adjust setting' metaphor in one program may be comfortable with it in another program:

P_1: " An icon containing the 'gear' sign is a good one for Settings, because it invokes the idea of a gear change on a bicycle" (Initial Conjecture)
O_1: "The 'gear' sign does not invoke the idea of a gear change on a bicycle *from my point of view*." (Counterexample)
P_2: "Ok, you're right, it does not invoke the idea of a gear change on a bicycle, but it is often used for Settings." (Monster-adjusting)

Example 4. Argumentation may also consider the role a given abstract design plays within a given icon set.

P_1: "Even without knowing what the first or third icon in Figure 8I stands for, I can make a conjecture that it has to do with a server or a user interface accessed via the cloud. However, with the second icon, I'm not sure what it means. It is composed of various signs that I don't understand. It's probably badly designed." (Conjecture)
O_1: "Did you notice that icons in Figure 8II and Figure 8III are both defined as part of the same icon set? They mean 'Stateful component' and 'Processing component' respectively. Therefore, the second icon is actually well designed, because it uses signs appearing in other icons of the same icon set." (Counterexample)

[2]From `http://cloudcomputingpatterns.org`.

P_2: "But the second icon contains a pipe sign that is not used anywhere within the icon set, so I still don't know what the second icon means. If there were an icon with a pipe sign with a clear meaning, then I could understand the second icon better. " (Strategic withdrawal)

The main characteristic of employing Lakatosian reasoning is that it allows a dynamic and social development of the intended meaning of blended icons. This cannot be achieved by using only abstract argumentation frameworks, since they assume that the object of discussion does not evolve. Therefore, having an argumentation process of this kind has several advantages: it promotes not only open-discussions around the meaning of an icon, but also the construction of a discourse about how an intended meaning is obtained.

This is a desirable characteristic in computational creativity when evaluating creative outcomes such as concept blends. In this way, the evaluation evolves into a refinement process of an initial created concept, giving much more flexibility at the moment of deciding whether a blend is suitable.

Discussion

We have illustrated the use of argumentation to evaluate completed blends. We alluded earlier to the role argumentation can play in the *generation* of blends, for instance by suggesting different ways to generalise the input spaces. Indeed, successive statements may serve to carry out the steps in the blending process iteratively, relaxing or refining as needed. These steps can be modelled using Lakatos's moves. From a conjectural candidate solution, to additional criteria that reveal this blend to be a 'monster' (i.e. which identify features of the candidate solution that cannot be allowed in the final solution for one reason or another), to adjustments that yield a more complete description of the problem and point the way toward a more satisfactory solution. An example of using argumentation for deciding which generalisations to use for creating a new icon is the following:

A: "We can create a different blend icon starting from the same icons of before." (see Figure 6)

B: "We could use the HARDDISK sign from the first icon and the DOCUMENT sign from the second icon."

A: "But putting the DOCUMENT sign above the HARDDISK does not make sense from my point of view."

B: "You're right, let's use the HARDDISK sign from the first icon and the PEN sign from the second icon."

A: "Sounds good, now we have a *Write-HardDisk* icon."

From this discussion and the previous sections, we think that it is feasible to bring the framework of argumentation inside the concept blending process. Moreover, this appears to work in a symmetric direction: the steps in an argumentation process can be carried out through blending. For instance, concept blending could be seen as the process behind the creation of rational arguments (Coulson and Pascual, 2006).

One area closely akin to the icon domain is the domain of *sentences* in a natural or artificial language. These can be evaluated for their coherence, succinctness, and fitness-to-purpose from a semantic standpoint (including relationship to other sentences), among other criteria; cf. (Abramsky and Sadrzadeh, 2014) for a category-theoretic view.

Since people have different standards for evaluation, they frequently disagree about what constitutes a satisfactory result, be it a final outcome or a design decision that is only a step the way to developing an artefact. They may also disagree at a more fundamental level about what can be considered a valid point of view or an appropriate manner of conducting an argument. For example, "Godwin's law" states that an online discussion ends when someone compares one of the discussants to Hitler and whoever made the comparison automatically loses the debate. Naturally, the validity of this principle is itself debatable. During the course of argumentation, the goalposts may shift, as new information is revealed about the domain under discussion, and about the discussants themselves. The relationship between argumentation and decision-making has been explored (Ouerdane, 2009), including the case of updating models of preferences (Ouerdane et al., 2014); the latter is quite similar to our previous work on Lakatos's games (Pease et al., 2014).

Conclusion and Future Work

Computational models of combinatorial creativity faces the daunting issue of evaluating a large number of possible novel combinations. Particularly, Fauconnier and Turner (1998) propose a model that includes a collection of optimality principles to guide the construction of a 'well-formed integration network'. Our computational model, based on generalisations of input spaces and amalgams, makes this combinatorial nature more explicit. The heuristic criteria called 'optimality principles' are too underspecified to be used as computational measures to evaluate and select possible blends. Moreover, alternative numeric measures may be not enough to evaluate the quality or novelty of creative artefacts. Our intuition is that in the context of creative outcomes, people use argumentation to understand, criticise, modify and evaluate them, and that computational argumentation is a useful tool for computational creativity.

The domain of computer icons generated by blending, where the evaluation of new icons is focused on their intended meaning, shows that symbolic argumentation is a process that is adequate to distinguish well-formed icons from mix-and-match combinations, unambiguous and clear icons from ambiguous or incomprehensible icons. This domain supports our claim that numeric heuristic evaluation measures are insufficient to recognise good blends, and shows the usefulness of an argumentation-based process for identifying good blends, detecting their critical problems, and refining them in an evolving, open-ended process.

We have shown how Lakatosian reasoning can be used in evaluating concept blending for icon design. Our approach offers two main advantages. Firstly, the evaluation process can *improve* the blend, since the dialogue about it refines resulting blends. Secondly, the *reasons behind* a particular evaluation are made explicit. This is crucial given recent work on the importance of context in creativity judgments (Charnley, Pease, and Colton, 2012; Colton, Pease, and Charnley, 2011). Argumentation offers a framing story that

shows how and why a particular artefact was constructed, which can be presented alongside the artefact itself.

We envision several future works. First, we intend to specify an ontology for modelling the semiotic system presented and to build a library of icons. Having a domain knowledge will allow us to generate arguments by induction, for instance, by analysing icons cases. Moreover, it will also open the possibility to explore the use of value-based argumentation (Bench-Capon, Doutre, and Dunne, 2002) for selecting the input icons to be used in the concept blending process. This latter point is important, since usually the inputs of a blending process are assumed to be already provided. Second, as far as the interpretation of icons is concerned, we are thinking to take advantage of existing approaches to natural language processing and understanding, especially Construction Grammars (CxG). In CxG, the grammatical construction is a pairing of form and content. In our semiotic system, sign patterns seem equivalent to the form, while interpretations would be akin to the content. Working with a grammar would make evaluation more explicit, e.g. we could use quantitative measures of ambiguity; and this would open many other domains for application.

Finally, we plan to implement Lakatosian reasoning by employing existing computational tools for argumentation (Devereux and Reed, 2010; Wells and Reed, 2012). Our goal is to provide a computational argumentation framework and to integrate it into the framework for computational creativity we are developing in the COINVENT project.

Acknowledgements

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

References

Abramsky, S., and Sadrzadeh, M. 2014. Semantic unification – A sheaf theoretic approach to natural language. In *Categories and Types in Logic, Language, and Physics*, volume 8222 of *LNCS*, 1–13. Springer.

Bench-Capon, T. J. M., and Dunne, P. E. 2007. Argumentation in artificial intelligence. *Artificial Intelligence* 171(10-15):619–641.

Bench-Capon, T. J. M.; Doutre, S.; and Dunne, P. E. 2002. Value-based argumentation frameworks. In *Artificial Intelligence*, 444–453.

Boden, M. A. 2003. *The Creative Mind - Myths and Mechanisms (2nd ed.)*. Routledge.

Bou, F.; Eppe, M.; Plaza, E.; and Schorlemmer, M. 2014. D2.1: Reasoning with Amalgams. Technical report, COINVENT Project. http://www.coinvent-project.eu/fileadmin/publications/D2.1.pdf.

Chandler, D. 2004. *Semiotics: The Basics*. Routledge.

Charnley, J.; Pease, A.; and Colton, S. 2012. On the notion of framing in computational creativiy. In *Proc. of the 3rd Int. Conf. on Computational Creativity*, 77–81.

Colton, S.; Pease, A.; and Charnley, J. 2011. Computational creativity theory: The FACE and IDEA descriptive models. In *2nd Int. Conf. on Computational Creativity*.

Coulson, S., and Pascual, E. 2006. For the sake of argument: Mourning the unborn and reviving the dead through conceptual blending. *Ann. Rev. of Cognitive Linguistics* 4:153–181.

Devereux, J., and Reed, C. 2010. Strategic argumentation in rigorous persuasion dialogue. In *ArgMAS*, volume 6057 of *LNCS*. Springer Berlin Heidelberg. 94–113.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77(2):321 – 357.

Falomir, Z.; Cabedo, L. M.; Abril, L. G.; Escrig, M. T.; and Ortega, J. A. 2012. A model for the qualitative description of images based on visual and spatial features. *Computer Vision and Image Understanding* 116(6):698–714.

Fauconnier, G., and Turner, M. 1998. Principles of conceptual integration. In Koenig, J. P., ed., *Discourse and Cognition: Bridging the Gap*. Center for the Study of Language and Information. 269–283.

Lakatos, I. 1976. *Proofs and refutations: the logic of mathematical discovery*. Cambridge University Press.

Ontañón, S., and Plaza, E. 2010. Amalgams: A formal approach for combining multiple case solutions. In *Proc. of the Int. Conf. on Case Base Reasoning*, volume 6176 of *LNCS*, 257–271. Springer.

Ouerdane, W.; Labreuche, C.; Maudet, N.; and Parsons, S. 2014. A dialogue game for recommendation with adaptive preference models. Technical report, Ecole Centrale Paris. Cahiers de recherche 2014-02.

Ouerdane, W. 2009. *Multiple Criteria Decision Aiding: a Dialectical Perspective*. Ph.D. Dissertation, University Paris-Dauphine, Paris, France.

Pease, A.; Budzynska, K.; Lawrence, J.; and Reed, C. 2014. Lakatos Games for Mathematical Argument. In *Proc. of COMMA*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, 59–66. IOS Press.

Prakken, H. 2005. Coherence and flexibility in dialogue games for argumentation. *J. Log. and Comput.* 15(6):1009–1040.

Rahwan, I., and Simari, G. R. 2009. *Argumentation in Artificial Intelligence*. Springer Publishing Company.

Schorlemmer, M.; Smaill, A.; Kühnberger, K.-U.; Kutz, O.; Colton, S.; Cambouropoulos, E.; and Pease, A. 2014. Coinvent: Towards a computational concept invention theory. In *5th Int. Conf. on Computational Creativity*.

Walton, D., and Krabbe, E. C. W. 1995. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press.

Wells, S., and Reed, C. 2012. A domain specific language for describing diverse systems of dialogue. *Journal of Applied Logic* 10(4):309 – 329.

A Process Model for Concept Invention

Roberto Confalonieri, Enric Plaza, Marco Schorlemmer
Artificial Intelligence Research Institute, IIIA-CSIC
Campus de la Universitat Autònoma de Barcelona (UAB)
E-08193 Bellaterra (Barcelona), Catalonia, Spain
{confalonieri,enric,marco}@iiia.csic.es

Abstract

In this paper, we propose a computational framework that models concept invention. The framework is based on conceptual blending, a cognitive theory that models human creativity and explains how new concepts are created. Apart from the blending mechanism modeling the creation of new concepts, the framework considers two extra dimensions such as origin and destination. For the former, we describe how a Rich Background supports the discovery of input concepts to be blended. For the latter, we show how arguments, promoting or demoting the values of an audience, to which the invention is headed, can be used to evaluate the candidate blends created. Throughout the paper, we exemplify the computational framework in the domain of computer icons.

Introduction

The cognitive theory of conceptual blending by Fauconnier and Turner (2002) models human creativity as a mental process according to which two input (mental) spaces are combined into a new mental space, called a blend. This theory, which was developed in the context of cognitive linguistics, posits that input mental spaces are somehow packaged by humans with the relevant information in the context in which the blend is created, and that blends are evaluated against some optimality principles (Fauconnier and Turner, 2002).

Existing computational models for concept invention — see the Related Work section for an overview— especially focus on the core mechanism of blending, that is, how blends are created, and re-interpret the optimality principles to evaluate the blends. In this position paper, we claim that a computational model also need to deal with two extra dimensions to which we refer as the *origin* and *destination* of concept invention. The origin considers from where and how input spaces are selected, whereas the destination considers to whom the creation is headed. These dimensions are justifiable if we think that there is no creation *ex nihilo* — thus, there is an origin — and there is usually a *purpose* in creating something new, and, consequently, there is a destination.

To this end, in this paper we propose to model concept invention by means of a process that consists of different sub-processes and components (Figure 1):

- **Rich Background and Discovery**: The origin consists of a Rich Background, the set of concepts available to be blended. This set is finite but complex, diverse, polymathic and heterogeneous. Concepts are associated with a background, understood as a person's education, experience, and social circumstances. The Rich Background supports a discovery process that finds pairs of concepts that can be blended.
- **Blending**: Conceptual blending is the mechanism according to which two concepts are combined into a blended concept. Blending is here characterised in terms of amalgams, a notion that was developed for combining cases in case-based reasoning (Ontañón and Plaza, 2010). Conceptual blending is modeled in terms of an amalgam-based workflow. The blending of two concepts may result in a large number of blends, that need to be evaluated.
- **Arguments, Values, Audiences and Evaluation**: Values are properties expected from a good blend. Values are considered as points of view and can be of different kinds, e.g., moral, aesthetic, etc. A destination or audience is characterised by a preference relation over these values. Arguments in favor or against a blend are built to evaluate the generated blends. An argument can promote or demote a value. In this way, the blends are evaluated depending on the audience for which they are created.

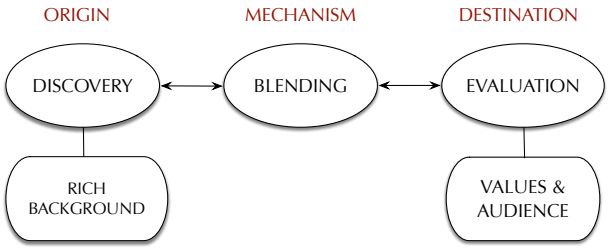


Figure 1: A process model for concept invention.

The above process model can be made more concrete in a domain such as *computer icon design*. In such a case, the Rich Background is what we can learn from, program about, specify of computer icons, such as a semiotic model of shapes, signs and relations between signs. This is understood as a finite and specific number of concepts given a particular set of icons (an icon library or a collection of libraries). Values, on the other hand, can be aesthetics such as simplicity or ambiguity, that matter for a specific type of

audience. These values serve to identify good icons that are created by the blending mechanism.

In the next section, we capture the process model above in terms of feature terms. This computational model is exemplified by means of a running example that shows the main processes that undergo the concept invention of new icons.

Related Work

Several approaches of formal and computational models for concept invention, inspired by the work of Fauconnier and Turner (2002), have been proposed.

Amalgam-based conceptual blending algorithms have been developed to blend CASL theories and $\mathcal{EL}^{++}$ concepts in (Confalonieri et al., 2015b; Eppe et al., 2015a,b). In these works, input spaces are assumed to be given. Good blends are selected by re-interpreting some optimality principles.

The Alloy algorithm for conceptual blending by Goguen and Harrell (2005) is based on the theory of algebraic semiotics (Goguen, 1999). Alloy has been integrated in the Griot system for automated narrative generation (Goguen and Harrell, 2005; Harrell, 2005, 2007). The input spaces of the Alloy algorithm are theories defined in the algebraic specification language OBJ (Malcolm, 2000). In the algorithm, input spaces are assumed to be given, hence there is no discovery. The optimality principles by Fauconnier and Turner (2002) are re-interpreted as *structural* optimality principles, and serve to prune the space of possible blends.

Sapper was originally developed by Veale and Keane (1997) as a computational model of metaphor and analogy. It computes a mapping between two separate domains — understood as graphs of concepts— that respects the relational structure between the concepts in each domain. Sapper can be seen as a computational model for conceptual blending, because the pairs of concepts that constitute its output can be manipulated as atomic units, as blended concepts (Veale and Donoghue, 2000). Strictly speaking, Sapper does not work with *a priori* given input spaces. It is the structure mapping algorithm itself which determines the set of concepts and relations between these concepts. In Sapper, most of the optimality principles are captured and serve to rank and filter the correspondences that comprise the mappings computed by the algorithm.

Divago, by Pereira (2007), is probably the first complete implementation of conceptual blending. The Divago's architecture includes different modules. A knowledge base contains different micro-theories and their instantiations. Of these, two are selected for the blending by the user or randomly, thus, no discovery is taken into account. A mapper then generates the generic space between the inputs, and passes it to a blender module which generates the 'blendoid', i.e., a projection that defines the space of possible blends. A factory component is used to select the best blends among the blendoid by means of a genetic algorithm. A dedicated module implements the optimality principles. Given a blend, this module computes a measure for each principle. These measures yield a preference value of the blend that is taken as the fitness value of the genetic algorithm.

Finally, another work that relates to ours is (Confalonieri et al., 2015a). The authors use Lakatosian reasoning to model dialogues in which users engage to discuss the intended meaning of an invented concept. The main difference with the current work relies on the way in which arguments are generated and used. Here, an argument is a reason for choosing a blend and it is generated automatically, whereas, in (Confalonieri et al., 2015a), an argument is a reason to refine the meaning of a blend and is provided by the user.

Computational Model

Rich Background

Let the Rich Background be a collection of computer icons. We assume that computer icons are described in terms of $form$ and a $meaning$. The form consists of a finite set of signs which are related by spatial relationships. Figure 2b(I) shows an example of an icon in which two signs, a MAGNIFYINGGLASS and a HARDDISK, are related by relation on. The meaning, on the other hand, is the interpretation that is given to an icon. For instance, a possible meaning associated to the icon in Figure 2b(I) is SEARCH-HARDDRIVE. We allow a sign to have different interpretations depending on the icons in which it is used.

We shall model the Rich Background by means of a finite set $\mathcal{C}$ of feature terms (Carpenter, 1992; Smolka and Aït-Kaci, 1989), each representing a concept. In this paper, feature terms are defined over a signature $\Sigma = \langle \mathcal{S}, \mathcal{F}, \preceq, \mathcal{X} \rangle$, where $\mathcal{S}$ is finite set of sort symbols, including $\top$ and $\bot$, which represent the most specific and the most general sort, respectively; $\mathcal{F}$ is a finite set of feature symbols; $\preceq$ is an order relation inducing an inheritance hierarchy such that $\bot \preceq s \preceq \top$, for all $s \in \mathcal{S}$; and $\mathcal{X}$ is a denumerable set of variables. Then, a feature term ψ has the form:

$$\psi := x : s[f_1 = \Psi_1, \ldots, f_n = \Psi_n]$$

with $n \geqslant 0$, and where $x \in \mathcal{X}$ is called the root variable of ψ (denoted as $\mathsf{root}(\psi)$), $s \in \mathcal{S}$ is the sort of x (denoted as $\mathsf{sort}(x)$), and, for all j with $1 \leqslant j \leqslant n$, $f_j \in \mathcal{F}$ are the features of x (denoted as $\mathsf{features}(x)$) and the values Ψ_j of the features are finite, non-empty sets of feature terms and/or variables (provided they are root variables of feature terms ocurring in ψ). When the set of values of a feature is a singleton set, we will omit the curly brackets in our notation. We will write $\mathsf{vars}(\psi)$ to denote the set of variables occurring in a feature term ψ.

We choose to model icons as concepts represented by feature terms over the signature with the following sort hierarchy $\mathcal{S}$:[1]

ICON
SIGN $\prec$ {ARROW, MAGNIFYINGGLASS, DOCUMENT, PEN, HARDDISK, CLOUD}
MEANING $\prec$ {ACTION, OBJECTTYPE}
ACTION $\prec$ {MODIFY, VIEWSEARCH, TRANSFER}
MODIFY $\prec$ {EDIT, WRITE}
VIEWSEARCH $\prec$ {SEARCH, FIND, ANALYSE}
TRANSFER $\prec$ {UPLOAD, DOWNLOAD}
OBJECTTYPE $\prec$ {INFOCONTAINER, DATACONTAINER}
INFOCONTAINER $\prec$ {PAGE, DOC, FILE}
DATACONTAINER $\prec$ {HARDDRIVE, CLOUD}

[1]The notation $s \prec \{s_1, \ldots, s_n\}$ denotes that $s_1, \ldots, s_n$ are sub-sorts of s.

$$x_1 : \text{ICON}\left[\begin{array}{l} form = \left\{\begin{array}{l} x_2 : \text{MAGNIFYINGGLASS}\left[\begin{array}{l} action = x_4 \\ on = x_3 \end{array}\right] \\ x_3 : \text{HARDDISK}\left[\ objectType = x_5\ \right] \end{array}\right\} \\ meaning = \left\{\begin{array}{l} x_4 : \text{SEARCH} \\ x_5 : \text{HARDDRIVE} \end{array}\right\} \end{array}\right]$$

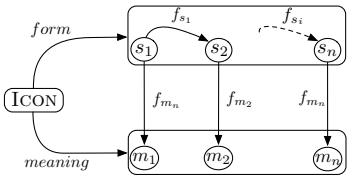
(a) Feature term representation of a computer icon.

(b) Examples of computer icons.

Figure 2: Rich Background about computer icons.

and features $\mathcal{F} = \{form, meaning, on, below, left, right, action, objectType\}$.

In addition, feature terms representing icons need to be of the following form. A representation of the structure of an icon is presented below and its description follows.



Root variables are of sort ICON and have at most two features $form$ and $meaning$, modelling the signs $(s_1, \ldots, s_n)$ and the meaning $(m_1, \ldots, m_n)$ of these signs in the context of the icon. Each sign is again represented by means of a feature term whose root variable is of sort $s \geq$ SIGN, and each meaning by means of feature terms whose root variable is of sort $s \geq$ MEANING.

Features of sign terms ($f_{s_1}, \ldots f_{s_n}$ in the schema above) are at most one of on, $left$, $right$, or $below$, specifying the spatial relationship between signs; and at most one of $action$ or $objectType$, specifying the meaning of signs ($f_{m_1}, \ldots f_{m_n}$ in the schema above). The values of spatial relation features are root variables of feature terms that are in the value of the $form$ feature; and those of features $action$ and $objectType$ are root variables of feature terms that are in the value of the $meaning$ feature. In addition the root variables in the value of the $action$ feature are of sort $s \geq$ ACTION, while those of the $objectType$ feature are of sort $s \geq$ OBJECTTYPE. Figure 2a shows the feature term representation of the icon in Figure 2b(I).

A fundamental relation between feature terms is that of subsumption ($\sqsubseteq$). Intuitively, a feature term ψ_1 subsumes another one ψ_2, or ψ_1 is more general than ψ_2, denoted as $\psi_1 \sqsubseteq \psi_2$, if all the information in ψ_1 is also in ψ_2.[2] We omit the formal definition of subsumption, which can be found in (Ontañón and Plaza, 2012) for feature terms as represented in this paper. The subsumption relation induces a partial order on the set of all features terms $\mathcal{L}$ over a given signature, that is, $\langle \mathcal{L}, \sqsubseteq \rangle$ is a poset.

[2]Notice that, in Description Logics, $A \sqsubseteq B$ has the inverse meaning "A is subsumed by B", since subsumption is defined from the set inclusion of the interpretations of A and B.

Discovery

In cognitive theories of conceptual blending, input spaces to be blended are givens that represent how humans package some relevant information in the context in which the blend is created.

In our computational model, an input space is a concept belonging to a library of concepts. The packaging of some relevant information corresponds to a discovery process that takes certain properties, which the blends need to satisfy, into account. In the creation of computer icons, we can imagine that an icon designer knows the meaning of an icon he wishes to create, but he ignores its form.

The discovery takes a query over the meaning of an icon concept as input, looks for concepts in the Rich Background, and returns an ordered set of pairs of concepts that can be blended. The query is modeled as a feature term ψ_q in which only the meaning part of an icon is specified. For instance, a query asking for an icon with meaning SEARCH-DOC is modeled as:

$$\psi_q := x_1 : \text{ICON}\left[\ meaning = \left\{\begin{array}{l} x_2 : \text{SEARCH} \\ x_3 : \text{DOC} \end{array}\right\}\ \right] \quad (1)$$

The matching of the query is not always a perfect match, since icon concepts in the Rich Background can have only one part of the meaning or similar meanings w.r.t. the meaning searched. To this end, the query resolution is modeled as a *similarity-based search*.

The main idea behind the similarity-based search is that, for each icon concept ψ_i in the Rich Background, we measure how ψ_q and ψ_i are similar and, we use this measure to rank the results. The similarity between two feature terms can be defined by means of their *anti-unification* or *Least General Generalisation* (LGG) (Ontañón and Plaza, 2012).

Definition 1 (Least General Generalisation) *The least general generalisation of two feature terms ψ_1 and ψ_2, denoted as $\psi_1 \sqcap \psi_2$, is defined as the most specific term that subsumes both:* $\psi_1 \sqcap \psi_2 = \{\psi \mid \psi \sqsubseteq \psi_1 \wedge \psi \sqsubseteq \psi_2 \wedge \nexists \psi' : \psi \sqsubset \psi' \wedge \psi' \sqsubseteq \psi_1 \wedge \psi' \sqsubseteq \psi_2\}$.

The least general generalisation encapsulates all the information that is common to both ψ_1 and ψ_2 and, for this reason, is relevant for defining a similarity measure.

The least general generalisation can be characterised as an operation over a refinement graph of feature terms. The refinement graph is derived from the poset $\langle \mathcal{L}, \sqsubseteq \rangle$ by means of a generalisation refinement operator γ.

$$\gamma(\psi) = \{\psi' \in \mathcal{L} \mid \psi' \sqsubseteq \psi \text{ and } \nexists \psi'' \text{ s.t. } \psi' \sqsubset \psi'' \sqsubset \psi\}$$

The above definition essentially says that γ is an operation that generalises a feature term to a set of feature terms that is an anti-chain. The refinement graph, then, is a directed graph whose nodes are feature terms, and for which there is an edge form feature term ψ_1 to ψ_2, whenever $\psi_2 \in \gamma(\psi_1)$. We shall call *generalisation paths* all finite paths $\psi \xrightarrow{\gamma} \psi'$ in a refinement graph, and denote with $\lambda(\psi \xrightarrow{\gamma} \psi')$ its length.

Ontañón and Plaza (2012) describe a generalisation operator for feature terms that consist of:

Sort generalisation, which generalises a term by substituting the sort of one of its variables by a more general sort;

Variable elimination, which generalises a term by removing the value of one of the features in one variables of the term (a variable is removed only when the variable does not have any features);

Variable equality elimination, which generalises a term by removing a variable equality and ensuring that $\perp$ can be reached from any term.

We refer to (Ontañón and Plaza, 2012) for the formal details of the operator.

It is worthy noticing that, in case of variable equalities, it is not possible to define a generalisation operator that finds all possible generalisations of a feature term. However, for the purpose of defining a least general generalisation-based similarity, an operator which ensures that $\perp$ is reachable in a finite number of steps will suffice.

Example 1 (LGG example) *Let us consider the feature terms ψ_q in Eq. 1 and ψ_1 in Figure 2a. The LGG $\psi_q \sqcap \psi_1$ is:*

$$x_1 : \text{ICON}\left[\ \textit{meaning} = \left\{ \begin{array}{l} x_2 = \text{SEARCH} \\ x_3 = \text{OBJECTTYPE} \end{array} \right\}\ \right]$$

$\psi_q \sqcap \psi_1$ captures the information shared among the icon concept ψ_1 and the query ψ_q. Both of them have two meanings. According to the ontology previously defined, the most general sorts for variables x_2 and x_3 are SEARCH *and* OBJECTTYPE *respectively. The* form *feature of ψ_1 is removed, since ψ_q does not contain this information.*

As previously said, the least general generalisation of two feature terms $\psi_1 \sqcap \psi_2$ is a symbolic representation of the information shared by ψ_1 and ψ_2. It can be used to measure the similarity between feature terms in a quantitative way. The refinement graph allows us to estimate the quantity of information of any feature term ψ. It is the length of the (minimal) generalisation path that leads from ψ to the most general term $\perp$. Therefore, the length $\lambda(\psi_1 \sqcap \psi_2 \xrightarrow{\gamma} \perp)$ estimates the informational content that is common to ψ_1 and ψ_2. In order to define a similarity measure, we need to compare what is common to ψ_1 and ψ_2 with what is not common. To this end, we take the lengths $\lambda(\psi_1 \xrightarrow{\gamma} \psi_1 \sqcap \psi_2)$ and $\lambda(\psi_2 \xrightarrow{\gamma} \psi_1 \sqcap \psi_2)$ into account. Then a similarity measure can be defined as follows.

Definition 2 (LGG-based similarity) *The LGG-based similarity between two feature terms ψ_1 and ψ_2, denoted by $S_\lambda(\psi_1, \psi_2)$, is:*

$$\frac{\lambda(\psi_1 \sqcap \psi_2 \xrightarrow{\gamma} \perp)}{\lambda(\psi_1 \sqcap \psi_2 \xrightarrow{\gamma} \perp) + \lambda(\psi_1 \xrightarrow{\gamma} \psi_1 \sqcap \psi_2) + \lambda(\psi_2 \xrightarrow{\gamma} \psi_1 \sqcap \psi_2)}$$

The measure S_λ estimates the ratio between the amount of information that is shared and the total information content. From a computational point of view, S_λ requires to compute two things. The LGG and the three lengths defined in the above equation. The algorithms for computing S_λ can be found in (Ontañón and Plaza, 2012).

Example 2 (Similarity example) *Let us consider the feature terms ψ_q in Eq. 1, ψ_1 in Figure 2a and their LGG in Example 1. Lengths $\lambda_1 = \lambda(\psi_1 \sqcap \psi_q \xrightarrow{\gamma} \perp) = 8$, $\lambda_2 = \lambda(\psi_1 \xrightarrow{\gamma} \psi_1 \sqcap \psi_q) = 12$, and $\lambda_3 = \lambda(\psi_q \xrightarrow{\gamma} \psi_1 \sqcap \psi_q) = 2$. Notice that λ_3 is very small (2 generalisations), while λ_2 is larger since ψ_1 has more generalised content. Therefore, the similarity between ψ_q and ψ_1 is:*

$$S_\lambda(\psi_1, \psi_q) = \frac{8}{12 + 2 + 8} = 0.36$$

$S_\lambda(\psi_1, \psi_q)$ expresses that these two concepts share the 36% of the total information.

Given the above definitions, the discovery of concepts can be implemented by a discovery algorithm. The algorithm accepts a Rich Background of concepts $\mathcal{C}$, a query ψ_q, and the generalisation operator γ as input, and returns a ranked set of pairs of concepts. This ranking can be done according to different strategies. One way is to build all pairs of concepts and rank them in a lexicographical order. The discovery returns a set of pairs of concepts $\langle (\psi_j, \lambda_j), (\psi_{j+1}, \lambda_{j+1}) \rangle$ in which $\lambda_j \geqslant \lambda_{j+1}$.

Blending

The computational model of concept blending is based on the notion of *amalgams* (Ontañón and Plaza, 2010). This notion was proposed in the context of case-based reasoning. Amalgams have also been used to model analogy (Besold and Plaza, 2015). According to this approach, input concepts are generalised until a generic space is found, and pairs of generalised input concepts are 'unified' to create blends.

Formally, the notion of amalgams can be defined in any representation language $\mathcal{L}$ for which a subsumption relation $\sqsubseteq$ between formulas (or descriptions) of $\mathcal{L}$ can be defined, together with an anti-unifier operation—playing the role of the generic space— and a unifier operation. Therefore, it can be defined for feature terms. We already defined the anti-unification of two feature term descriptions (Definition 1). Now, we proceed to define their unification.

Definition 3 (Unification) *The* unification *of two feature terms ψ_1 and ψ_2, denoted as $\psi_1 \sqcup \psi_2$, is defined as the most general term that is subsumed by both: $\psi_1 \sqcup \psi_2 = \{\psi \mid \psi_1 \sqsubseteq \psi \wedge \psi_2 \sqsubseteq \psi \ \wedge \nexists \psi' : \psi' \sqsubset \psi \wedge \psi_1 \sqsubseteq \psi' \wedge \psi_2 \sqsubseteq \psi'\}$.*

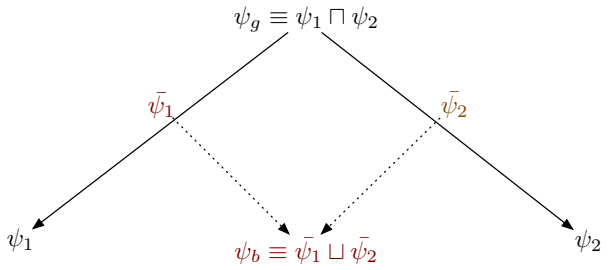


Figure 3: A diagram of a blend ψ_b from inputs ψ_1 and ψ_2.

Intuitively, a unifier is a description that has all the information in both the original descriptions. If joining this information leads to inconsistency, this is equivalent to say that $\psi_1 \sqcup \psi_2 = \top$, e.g., they have no common specialisation except 'none'.

An *amalgam* or *blend* of two descriptions is a new description that contains *parts from these two descriptions*. For instance, an amalgam of 'a red French sedan' and 'a blue German minivan' is 'a red German sedan'; clearly, there are always multiple possibilities for amalgams, like 'a blue French minivan'.

For the purposes of this paper, we define an *amalgam* or *blend* of two input descriptions as follows:

Definition 4 (Blend) *A description $\psi_b \in \mathcal{L}$ is a blend of two inputs ψ_1 and ψ_2 (with LGG $\psi_g = \psi_1 \sqcap \psi_2$) if there exist two generalisations $\bar{\psi}_1$ and $\bar{\psi}_2$ such that: 1) $\psi_g \sqsubseteq \bar{\psi}_1 \sqsubseteq \psi_1$, 2) $\psi_g \sqsubseteq \bar{\psi}_2 \sqsubseteq \psi_2$, and 3) $\psi_b \equiv \bar{\psi}_1 \sqcup \bar{\psi}_2 \not\equiv \top$.*

The above definition is illustrated in Figure 3, where the LGG of the inputs is indicated as ψ_g, and the blend ψ_b is the unification of two concrete generalisations $\bar{\psi}_1$ and $\bar{\psi}_2$ of the inputs. Equality ($\equiv$) here should be understood as $\sqsubseteq$-equivalence, that is, $\psi_1 \equiv \psi_2$ iff $\psi_1 \sqsubseteq \psi_2$ and $\psi_2 \sqsubseteq \psi_1$.

Usually one is interested only in maximal blends, e.g., in those blends that contain the maximal information of their inputs. A blend ψ_b of two inputs ψ_1 and ψ_2 is maximal if there is no other blend ψ'_b of ψ_1 and ψ_2 such that $\psi_b \sqsubset \psi'_b$. The reason why one is interested in maximal blends is that a maximal blend captures as much information as possible from the inputs. Moreover, any non-maximal blend can be obtained by generalising a maximal blend.

However, the number of blends that satisfies the above definition can still be very large and selection criteria for filtering and ordering them are therefore needed. Fauconnier and Turner (2002) discuss optimality principles, however, these principles are difficult to capture in a computational way, and other selection strategies need to be explored.

We interpret blend evaluation in two steps. First, we discard those blends that do not satisfy a query ψ_q. Then, we order the blends by means of arguments, values and audiences in order to decide which blend is the best one.

Arguments, Values and Audiences

An argument is a central notion in several models for reasoning about defeasible information (Dung, 1995; Pollock, 1992), decision making (Amgoud and Prade, 2009; Bonet and Geffner, 1996), practical reasoning (Atkinson, Bench-Capon, and McBurney, 2004), and modeling different types of dialogues such as persuasion (Bench-Capon, 2003). In most existing works on argumentation, an argument is a reason for believing a statement, choosing an option, or doing an action. Depending on the application domain, an argument is either considered as an abstract entity whose origin and structure are not defined, or it is a logical proof for a statement where the proof is built from a knowledge base.

In our model, arguments are reasons for accepting or rejecting a given blend. They are built by the agent when calculating the different values associated with a blend. Values are considered as points of view and can have different origins, e.g., they can be moral, aestethic, etc.

Generally, there can be several values $\mathcal{V} = \{v_1, \ldots, v_k\}$. Each value is associated with a degree that belongs to the scale $\Delta = (0, \ldots, 1]$, where 0 and 1 are considered the worst and the best degree respectively. For our purposes, we will consider values such as *simplicity* and *unambiguity*.

The main idea behind simplicity is that we want to estimate how simple an icon is from a representation point of view. This can be done by counting the quantity of information used in the feature term describing an icon. We can assume that simple icons are those described with less information. Therefore, simplicity is defined to be inversely proportional to the total number of features and sorts used in the variables of a feature term ψ_b.

$$\mathsf{Simplicity}(\psi_b) = \frac{1}{\sum_{x \in \mathsf{vars}(\psi_b)} \mathsf{features}(x) + \mathsf{sorts}(x)}$$

Unambiguity, on the other hand, measures how many interpretations an icon has w.r.t. the Rich Background. Since icons are polysemic —they can be interpreted in different ways— there can be icons that contain the same sign but the sign is associated with a different meaning. To define the unambiguity value, let us first define the polysemic set of ψ_b as:

$$\begin{aligned}\mathsf{Pol}(\psi_b) = \{&\psi_j \in \mathcal{C} \mid \exists s \in \mathsf{form}(\psi_j) \cap \mathsf{form}(\psi_b) \\ &\wedge \mathsf{meaning}(\psi_j, s) \neq \mathsf{meaning}(\psi_b, s)\}\end{aligned}$$

where $\mathsf{form}(\psi_j)$ is a function that returns the value of feature *form*, i.e., the set of signs used in the icon represented by feature term ψ_j; and $\mathsf{meaning}(\psi_j, s)$ is a function that returns the sort of the variable that is the value of feature *action* or *objectType* of the variabe of sort s, i.e., the meaning used for the sign represented by sort s in feature term ψ_j. Then, the unambiguity value is defined to be inversely proportional to the cardinality of Pol.

$$\mathsf{Unambiguity}(\psi_b) = \begin{cases} 1/|\mathsf{Pol}(\psi_b)| & \text{if } |\mathsf{Pol}(\psi_b)| \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

Values play a different role depending on the target or audience towards which the creation is headed. Audiences are characterised by the values and by a preferences among these values. Given a set of values $\mathcal{V}$, there are potentially as many audiences as there are orderings on $\mathcal{V}$.

Definition 5 (Audience) *An audience is a binary relation $\mathcal{R} \subseteq \mathcal{V} \times \mathcal{V}$ which is irreflexive, asymmetric, and transitive.*

We say that v_i is preferred to v_j in the audience $\mathcal{R}$, denoted as $v_i \succ_{\mathcal{R}} v_j$, if $\langle v_i, v_j \rangle \in \mathcal{R}$. We say that a value v_j covers v_i in the audience $\mathcal{R}$, denoted as $v_i \succcurlyeq_{\mathcal{R}} v_j$, if $v_i \succ_{\mathcal{R}} v_j$ and $\nexists v_{i'}$ such that $v_i \succ_{\mathcal{R}} v_{i'} \succ_{\mathcal{R}} v_j$.

Given a blend, an argument is generated for each value. The degree of the value characterises the 'polarity' of the argument which can be *pro* or *con* a blend. Arguments pros promote a blend whereas arguments cons demote it. Given a set of blends $\mathcal{B}$, the tuple $\langle \mathcal{B}, \mathcal{V}, \Delta \rangle$ will be called a theory.

Definition 6 (Argument) *Let $\langle \mathcal{B}, \mathcal{V}, \Delta \rangle$ be a theory.*

- *An* argument pro *a blend b is a tuple $\langle (v, \delta), b \rangle$ where $v \in \mathcal{V}$, $\delta \in \Delta$ and $0.5 \leqslant \delta \leqslant 1$*
- *An* argument con *b is a pair $\langle (v, \delta), b \rangle$ where $v \in \mathcal{V}$, $\delta \in \Delta$ and $0 < \delta < 0.5$*

A function Val *returns the value v associated with an argument and a function* Deg *returns δ.*

The blend evaluation can be formulated as a decision problem in which one has to decide an order relation $\succeq_{\mathcal{B}}$ on the set of candidate blends $\mathcal{B}$. The definition of this relation is based on the set of arguments pros and cons associated with the candidate blends. Depending on the kind of arguments that are considered and how they are handled, different decision criteria can be defined (Amgoud and Prade, 2009):

- **Unipolar decision criteria:** they focus either only on arguments pros or arguments cons;
- **Bipolar decision criteria:** they take both arguments pros and cons into account;
- **Meta-criteria:** they aggregate arguments pros and cons into a meta-argument.

In what follows, we denote the set of arguments pros and cons as $\mathcal{A}_p = \{\alpha_1, \ldots, \alpha_n\}$ and $\mathcal{A}_c = \{\alpha_1, \ldots, \alpha_m\}$ respectively. Besides, we assume to have the following functions: $\mathcal{M}_p : \mathcal{B} \rightarrow 2^{\mathcal{A}_p}$ and $\mathcal{M}_c : \mathcal{B} \rightarrow 2^{\mathcal{A}_c}$ that return the set of arguments pros and the set of arguments cons associated with a blend respectively; $\mathcal{M} : \mathcal{B} \rightarrow 2^{\mathcal{A}_p \cup \mathcal{A}_c}$ that returns all arguments associated with a blend.

A basic decision criterion for comparing candidate blends can be defined by comparing the number of arguments pros associated with them.

Definition 7 *Let $b_1, b_2 \in \mathcal{B}$. $b_1 \succeq_{\mathcal{B}} b_2$ if and only if $|\mathcal{M}_p(b_1)| \geqslant |\mathcal{M}_p(b_2)|$.*

Notice that the above criterion guarantees that any pair of blends can be compared.

When the audience is taken into account, one may think of preferring a blend that has an argument pro whose value is preferred to the values of any argument pro the other blends.

Definition 8 *Let $b_1, b_2 \in \mathcal{B}$. $b_1 \succeq_{\mathcal{B}} b_2$ if and only if $\exists \alpha \in \mathcal{M}_p(b_1)$ such that $\forall \alpha' \in \mathcal{M}_p(b_2)$, $\mathsf{Val}(\alpha) \succ_{\mathcal{R}} \mathsf{Val}(\alpha')$.*

In the above definition, $\succeq_{\mathcal{B}}$ depends on the relation $\succ_{\mathcal{R}}$. Since $\succ_{\mathcal{R}}$ is a preference relation, some of the values of the arguments can be incomparable. Consequently, b_1 and b_2 will not be comparable neither. This definition can be relaxed, for instance, by ignoring these arguments.

The counter-part decision criteria of Definitions 7-8 for the case of arguments cons can be defined in a similar way and we omit them.

In the case of bipolar decision criteria, we can combine the criterion dealing with arguments pros with the criterion dealing with arguments cons.

Definition 9 *Let $b_1, b_2 \in \mathcal{B}$. $b_1 \succeq_{\mathcal{B}} b_2$ if and only if $|\mathcal{M}_p(b_1)| \geqslant |\mathcal{M}_p(b_2)|$ and $|\mathcal{M}_c(b_1)| \leqslant |\mathcal{M}_c(b_2)|$.*

Unfortunately, the above definition does not ensure that we can compare all the blends.

Finally, meta-criteria for deciding which blends are preferred can be defined by aggregating arguments pros and cons into a meta-argument. Then, comparing two blends amounts to compare the resulting meta-arguments. A simple criterion can be defined by aggregating the degrees of the arguments associated with a blend.

Definition 10 *Let $b_1, b_2 \in \mathcal{B}$. $b_1 \succeq_{\mathcal{B}} b_2$ if and only if*

$$\sum_{\alpha \in \mathcal{M}(b_1)} \mathsf{Deg}(\alpha) \geqslant \sum_{\alpha' \in \mathcal{M}(b_2)} \mathsf{Deg}(\alpha')$$

This definition can be extended to take the audience into account. To this end, we consider a rank function that maps each value of $\mathcal{R}$ to an integer. The rank function is defined as follows:

$$\mathsf{Rank}_{\mathcal{R}}(v) = \begin{cases} 1 & \text{if } \nexists v' \text{ s.t. } v' \succcurlyeq_{\mathcal{R}} v \\ \max\limits_{v' \succcurlyeq_{\mathcal{R}} v} \{\mathsf{Rank}_{\mathcal{R}}(v')\} + 1 & \text{otherwise} \end{cases}$$

Essentially, Rank counts how many values a certain value covers. This ranking is then used to define the following audience-based aggregation decision criterion.

Definition 11 *Let $b_1, b_2 \in \mathcal{B}$. $b_1 \succeq_{\mathcal{B}} b_2$ if and only if*

$$\sum_{\alpha \in \mathcal{M}(b_1)} \frac{\mathsf{Deg}(\alpha)}{\mathsf{Rank}_{\mathcal{R}}(\mathsf{Val}(\alpha))} \geqslant \sum_{\alpha' \in \mathcal{M}(b_2)} \frac{\mathsf{Deg}(\alpha')}{\mathsf{Rank}_{\mathcal{R}}(\mathsf{Val}(\alpha'))}$$

This definition also guarantees that all the blends are comparable.

The Model at Work

Let us imagine an agent that has access to a Rich Background $\mathcal{C} = \{\psi_1, \psi_2, \psi_3, \psi_4\}$ consisting of four of the icons depicted in Figures 2b(I-II-III-IV). As previously described, ψ_1 is a feature term representing an icon with meaning SEARCH-HARDDISK. ψ_2 represents an icon that consists of two sorts of type sign, an ARROW and a CLOUD, whose meaning is DOWNLOAD-CLOUD. ψ_3 represents an icon with two sorts of type sign, a PEN and a DOCUMENT, whose meaning is EDIT-DOC; finally, ψ_4 is a feature term that consists of three sorts, ARROW, DOCUMENT and CLOUD with the intended meaning of DOWNLOAD-DOC-CLOUD.

The agent receives as input a query asking for an icon with meaning SEARCH-DOC, ψ_q (Eq. 1), and an audience, that is, a preference order over the values. For the sake of this example, we assume that $\mathsf{Simplicity} \succ_{\mathcal{R}} \mathsf{Unambiguity}$.

The discovery retrieves the following pairs of concepts:

$$\{\langle (\psi_1, 0.36), (\psi_3, 0.36) \rangle\}, \{\langle (\psi_1, 0.36), (\psi_2, 0.27) \rangle\}$$

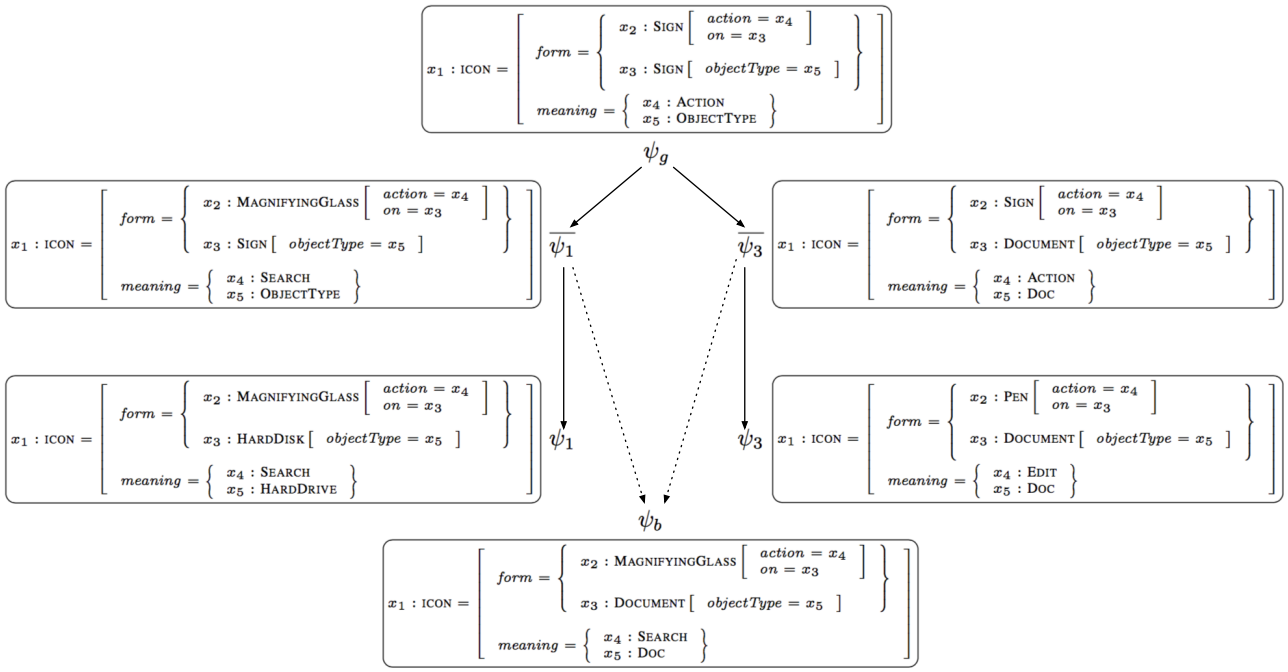


Figure 4: Amalgam-based blending of feature terms ψ_1 and ψ_3.

$\{\langle(\psi_3, 0.36), (\psi_2, 0.27)\rangle\}, \{\langle(\psi_1, 0.36), (\psi_4, 0.25)\rangle\}$

$\{\langle(\psi_3, 0.36), (\psi_4, 0.25)\rangle\}, \{\langle(\psi_2, 0.27), (\psi_4, 0.25)\rangle\}$

The agent proceeds to blend the first pair in the list. To this end, it applies the amalgam-based blending. The least general generalisation of ψ_1 and ψ_3 is an icon with two sorts of type SIGN, one *on* the other one, and with meaning ACTION and OBJECTTYPE respectively. The agents explores the space of generalisations and finds two maximal blends; a blend ψ_{b_1} describing an icon with two sorts of type MAGNIFYINGGLASS and DOCUMENT whose meaning is SEARCH-DOC; another blend ψ_{b_2} describing an icon with sorts of type PEN and HARDDISK whose meaning is EDIT-HARDDRIVE. Since ψ_{b_2} does not satisfy the query, is discarded, and only ψ_{b_1} is kept. The creation of ψ_{b_1} is illustrated in Figure 4.

The agents repeats the above procedure for each pair discovered. Finally, it finds another blend, which satisfies ψ_q, by blending the pair ψ_1 and ψ_4. It is a blend describing an icon with three sorts of type MAGNIFYINGGLASS, DOCUMENT, and CLOUD whose meaning is SEARCH-DOC-CLOUD. Intuitively, this blend can be obtained by generalising HARDDISK from ψ_1 and ARROW from ψ_4, and by keeping the other input icons' specifics. We denote this blend as ψ_{b_2}. The set of blends is $\mathcal{B} = \{\psi_{b_1}, \psi_{b_2}\}$. A representation of ψ_{b_1} and ψ_{b_2} is given in Figures 2b(V-VI).

The agent evaluates these blends by means of the arguments and values described in the previous section. The blend ψ_{b_1} contains 10 variables whereas ψ_{b_2} contains 14. Therefore, the simplicity value's degrees of ψ_{b_1} and ψ_{b_2} are 0.1 and 0.07 respectively. Their unambiguity, on the other hand, is 1, since the Rich Background does not contain icons with the same signs used in ψ_{b_1} and ψ_{b_2}, but with a different meaning. The arguments built by the agent are:

	Simplicity	Unambiguity
ψ_{b_1}	0.1	1
ψ_{b_2}	0.07	1

Therefore, both blends have an argument pro regarding their simplicity and an argument con w.r.t. their unambiguity value. It is easy to see that the blends are ranked in different ways when using the criteria we defined. For instance, ψ_{b_1} and ψ_{b_2} are equally preferred when counting their arguments pros (or cons) (Definition 7), and when considering both arguments pros and cons (Definition 9). Instead, ψ_{b_1} is preferred to ψ_{b_2} when using the criteria that take the audience into account (Definitions 8 and 11).

Conclusion and Future Work

In this paper, we described a process model for concept invention that is based on and extends the conceptual blending theory of Fauconnier and Turner (2002). According to this process, concept invention is characterised by different sub-processes—discovery, blending, and evaluation—that together account for concept invention. We proposed its computational model in terms of feature terms, a formal knowledge representation language. This allowed us to capture the concept invention process in terms of well-defined operators such as anti-unification—for computing a generic space—and unification—for computing a blend. Pairs of input concepts are retrieved from a Rich Background by means of a discovery process that takes a similarity measure into account. Blending is realised according to the notion

of amalgam, and blend evaluation is achieved by means of arguments, values and audience.

We exemplified the computational framework in the domain of computer icon design, but the framework is general enough to be used in other domains such as music or poetry generation. We plan to explore the use of arguments, values and audiences as a means to evaluate concept blends in such domains as future work.

We also aim at extending the process model by including the notion of coherence by Thagard (2000). Coherence theory, when used to explain human reasoning, proposes that humans accept or reject a cognition depending on how much it contributes to maximising the constraints imposed by situations or other cognitions. In the case of concept invention, coherence can be defined and used, for instance, to measure to what extent a blend coheres or incoheres with the Rich background and other blends.

Acknowledgments

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

References

Amgoud, L., and Prade, H. 2009. Using arguments for making and explaining decisions. *Artificial Intelligence* 173:413–436.

Atkinson, K.; Bench-Capon, T.; and McBurney, P. 2004. Justifying practical reasoning. In *Proc. of the Fourth Workshop on Computational Models of Natural Argument (CMNA'04)*, 87–90.

Bench-Capon, T. J. M. 2003. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation* 13(3):429–448.

Besold, T. R., and Plaza, E. 2015. Generalize and Blend: Concept Blending Based on Generalization, Analogy, and Amalgams. In *Proc. of the 6th Int. Conf. on Computational Creativity, ICCC15*.

Bonet, B., and Geffner, H. 1996. Arguing for decisions: A qualitative model of decision making. In *Proc. of the 12th Conf. on Uncertainty in Artificial Intelligence (UAI'96)*, 98–105.

Carpenter, B. 1992. *The Logic of Typed Feature Structures*. New York, NY, USA: Cambridge University Press.

Confalonieri, R.; Corneli, J.; Pease, A.; Plaza, E.; and Schorlemmer, M. 2015a. Using Argumentation to Evaluate Concept Blends in Combinatorial Creativity. In *Proc. of the 6th Int. Conf. on Computational Creativity, ICCC15*.

Confalonieri, R.; Eppe, M.; Schorlemmer, M.; Kutz, O.; Peñaloza, R.; and Plaza, E. 2015b. Upward Refinement for Conceptual Blending in Description Logic —An ASP-based Approach and Case Study in $\mathcal{EL}^{++}$. In *Proc. of 1st Int. Workshop of Ontologies and Logic Programming for Query Answering.* Co-located with IJCAI-2015.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77:321–357.

Eppe, M.; Confalonieri, R.; Maclean, E.; Kaliakatsos-Papakostas, M. A.; Cambouropoulos, E.; Schorlemmer, W. M.; Codescu, M.; and Kühnberger, K. 2015a. Computational Invention of Cadences and Chord Progressions by Conceptual Chord-Blending. In *IJCAI 2015*, 2445–2451.

Eppe, M.; Maclean, E.; Confalonieri, R.; Kutz, O.; Schorlemmer, W. M.; and Plaza, E. 2015b. ASP, Amalgamation, and the Conceptual Blending Workflow. In *Logic Programming and Nonmonotonic Reasoning - 13th Int. Conf., LPNMR 2015, Lexington, KY, USA, September 27-30, 2015.*, 309–316.

Fauconnier, G., and Turner, M. 2002. *The Way We Think: Conceptual Blending And The Mind's Hidden Complexities*. Basic Books.

Goguen, J. A., and Harrell, D. F. 2005. Foundations for active multimedia narrative: Semiotic spaces and structural blending. Manuscript to appear published in the journal "Interaction Studies: Social Behaviour and Communication in Biological and Artificial Systems".

Goguen, J. 1999. An introduction to algebraic semiotics, with application to user interface design. In Nehaniv, C. L., ed., *Computation for Metaphors, Analogy, and Agents*, volume 1562 of *LNCS*. 242–291.

Harrell, D. F. 2005. Shades of computational evocation and meaning: The GRIOT system and improvisational poetry generation. *6th Digital Arts and Culture Conference*.

Harrell, F. 2007. *Theory and technology for computational narrative: an approach to generative and interactive narrative with bases in algebraic semiotics and cognitive linguistics*. Ph.D. Dissertation, University of California, San Diego.

Malcolm, G. 2000. *Software Engineering with OBJ: algebraic specification in action*. Kluwer.

Ontañón, S., and Plaza, E. 2012. Similarity measures over refinement graphs. *Machine Learning* 87(1):57–92.

Ontañón, S., and Plaza, E. 2010. Amalgams: A Formal Approach for Combining Multiple Case Solutions. In *Proc. of the Int. Conf. on Case Base Reasoning*, volume 6176 of *Lecture Notes in Computer Science*, 257–271. Springer.

Pereira, F. C. 2007. *Creativity and Artificial Intelligence: A Conceptual Blending Approach*. Mouton de Gruyter.

Pollock, J. 1992. How to reason defeasibly. *Artificial Intelligence Journal* 57:1–42.

Smolka, G., and Aït-Kaci, H. 1989. Inheritance hierarchies: Semantics and unification. *Journal of Symbolic Computation* 7(3–4):343—370.

Thagard, P. 2000. *Coherence in thought and action*. The MIT Press.

Veale, T., and Donoghue, D. O. 2000. Computation and blending. *Cognitive Linguistics* 11(3-4):253–282.

Veale, T., and Keane, M. 1997. The competence of sub-optimal theories of structure mapping on hard analogies. In *IJCAI*, 232–237.

Coherent Concept Invention

Marco Schorlemmer, Roberto Confalonieri, and Enric Plaza

Artificial Intelligence Research Institute, IIIA-CSIC
Bellaterra (Barcelona), Catalonia, Spain
{marco,confalonieri,enric}@iiia.csic.es

Abstract. We address the problem on how newly invented concepts are evaluated with respect to a background ontology of conceptual knowledge so as to decide which of them are to be accepted into a system of familiar concepts, and how this, in turn, may affect the previously accepted conceptualisation. As technique to tackle this problem we explore the applicability of Paul Thagard's computational theory of coherence. In particular, we propose a formalisation of Thagard's notion of *conceptual coherence* for concepts represented in the $\mathcal{AL}$ description logic and explore by means of an illustrative example the role coherence may play in the process of conceptual blending.

Keywords: conceptual blending, coherence, description logics

1 Introduction

Combinational creativity —when novel ideas (concepts, theories, solutions, works of art) are produced through unfamiliar combinations of familiar ideas— is, of the three forms of creativity put forward by Boden, the most difficult to capture computationally [2]. Putting concepts together to generate new concepts is, in principle, not a difficult task; but doing this in a computationally tractable way, and being able to recognise the value of newly invented concepts for better understanding a certain domain, is not as straightforward.

An important recent development that has significantly influenced the current understanding of the general cognitive principles operating during concept invention is Fauconnier and Turner's theory of *conceptual blending* [6, 7]. Fauconnier and Turner proposed conceptual blending as the fundamental cognitive operation underlying much of everyday thought and language, and modelled it as a process by which humans subconsciously combine particular elements and their relations of originally separate conceptual spaces into a unified space, in which new elements and relations emerge, and new inferences can be drawn.

The theory has been primarily applied as an analytic tool for describing already existing blends of ideas and concepts in a varied number of fields, such as linguistics, music theory, poetics, mathematics, theory of art, political science, discourse analysis, philosophy, anthropology, and the study of gesture and of material culture [20]. But it has been also widely recognised to be a theory that can serve as a basis for computational models of creativity [5, 9, 10, 15, 21].

To guide the concept invention process, in addition to the blending mechanism *per se*, at least two additional dimensions need to be considered, namely the origin and destination of concept invention, i.e., from where (and how) input concepts are selected and to whom the concept invention is headed. Confalonieri et al. have proposed a process model for concept invention in which these dimensions are taken into account [3]. Inputs are selected based on a similarity measure that is computed relative to a Rich Background, and blends are evaluated using an argumentation framework based on value preferences of the audience for which concepts are invented.

In this paper, we aim at showing how Thagard's computational theory of coherence [19] could also serve as an additional mechanism for triggering concept invention and evaluating newly blended concepts. In [18], Thagard suggested to use coherence as a model for the closely related cognitive process of *conceptual combination*, where the focus is primarily on language compositionality such as noun-noun or adjective-noun combinations [17]. Kunda and Thagard, for instance, show how conceptual coherence can be used for describing how we reason with social stereotypes [12].

Building upon Thagard's intuitions and principles for modelling coherence, we propose a formalisation of Thagard's notion of conceptual coherence for concepts represented in a description logic —we take the basic description logic $\mathcal{AL}$ as a start— and further explore its applicability to conceptual blending. But instead of interpreting coherence or incoherence based on statistical correlations or causal relations (i.e., on frequencies of positive or negative association), we determine coherence and incoherence as dependent on how concept descriptions are stated. Failure to find conceptual blends that cohere with some given background knowledge leads to a search for alternative conceptual blends that eventually increase the overall coherence of the blend with the background knowledge.

The paper is organised as follows: In Section 2 we give a brief overview of Thagard's computational theory of coherence, in Section 3 we introduce some core definitions regarding coherence and coherence graphs, and in Section 4 we provide a formalisation of conceptual coherence for the description logic $\mathcal{AL}$. Conceptual blending in $\mathcal{AL}$ is described in Section 5, and coherence is applied to blending in Section 6. We conclude in Section 7.

2 Thagard's Computational Theory of Coherence

Thagard addresses the problem of determining which pieces of information, such as hypotheses, beliefs, propositions or concepts, to accept and which to reject based on how they cohere and incohere among them, given that, when two elements cohere, they tend to be accepted together or rejected together; and when two elements incohere, one tends to be accepted while the other tends to be rejected [19].

This can be reformulated as a constraint satisfaction problem as follows. Pairs of elements that cohere between them form positive constraints, and pairs of elements that incohere between them form negative constraints. If we partition

the set of pieces of information we are dealing with into a set of accepted elements and a set of rejected elements, then a positive constraint is satisfied if both elements of the constraint are either among the accepted elements or among the rejected ones; and a negative constraint is satisfied if one element of the constraint is among the accepted ones and the other is among the rejected ones. The coherence problem is to find the partition that maximises the number of satisfied constraints.

Note that in general we may not be able to partition a set of elements as to satisfy *all* constraints, thus ending up accepting elements that incohere between them or rejecting an element that coheres with an accepted one. The objective is to minimise these undesired cases. The coherence problem is known to be NP-complete, though there exist algorithms that find good enough solutions of the coherence problem while remaining fairly efficient.

Depending on the kind of pieces of information we start from, and on the way the coherence and incoherence between these pieces of information is determined, we will be dealing with different kinds of coherence problems. So, in *explanatory coherence* we seek to determine the acceptance or rejection of hypotheses based on how they cohere and incohere with given evidence or with competing hypotheses; in *deductive coherence* we seek to determine the acceptance of rejection of beliefs based on how they cohere and incohere due to deductive entailment or contradiction; in *analogical coherence* we seek to determine the acceptance or rejection of mapping hypotheses based on how they cohere or incohere in terms of structure; and in *conceptual coherence* we seek to determine the acceptance or rejection of concepts based on how they cohere or incohere as the result of the positive or negative associations that can be established between them. Thagard discusses these and other kinds of coherence.

Although Thagard provides a clear technical description of the coherence problem as a constraint satisfaction problem, and he enumerates concrete principles that characterise different kinds of coherences, he does not clarify the actual nature of the coherence and incoherence relations that arise between pieces of information, nor does he suggest a precise formalisation of the principles he discusses. Joseph et al. have proposed a concrete formalisation and realisation of deductive coherence [11], which they applied to tackle the problem of norm adoption in normative multi-agent system. In this paper, we shall focus on the problem of conceptual coherence and its applicability to conceptual blending.

3 Preliminaries: Coherence Graphs

In this section we give precise definitions of the concepts intuitively introduced in the previous section.

Definition 1. *A* coherence graph *is an edge-weighted, undirected graph* $G = \langle V, E, w \rangle$*, where:*

1. *V is a finite set of nodes representing pieces of information.*

2. $E \subseteq V^{(2)}$ *(where* $V^{(2)} = \{\{u, v\} \mid u, v \in V\}$*) is a finite set of edges representing the coherence or incoherence between pieces of information.*
3. $w : E \to [-1, 1] \setminus \{0\}$ *is an edge-weighted function that assigns a value to the coherence between pieces of information.*

Edges of coherence graphs are also called constraints.

When we partition the set V of vertices of a coherence graph (i.e., the set of pieces of information) into a set A of accepted elements and a set $R = V \setminus A$ of rejected elements, then we can say when a constraint —an edge between vertices— is satisfied or not by the partition.

Definition 2. *Given a coherence graph* $G = \langle V, E, w \rangle$*, and a partition* (A, R) *of* V*, the set of* satisfied constraints $C_{(A,R)} \subseteq E$ *is given by:*

$$C_{(A,R)} = \left\{ \{u, v\} \in E \;\middle|\; \begin{array}{l} u \in A \text{ iff } v \in A, \text{ whenever } w(\{u, v\}) > 0 \\ u \in A \text{ iff } v \in R, \text{ whenever } w(\{u, v\}) < 0 \end{array} \right\}$$

All other constraints (i.e., those in $E \setminus C_{(A,R)}$*) are said to be* unsatisfied.

The coherence problem is to find the partition of vertices that satisfies as much constraints as possible, i.e., to find the partition that maximises the coherence value as defined as follows, which makes coherence to be independent of the size of the coherence graph.

Definition 3. *Given a coherence graph* $G = \langle V, E, w \rangle$*, the* coherence of a partition (A, R) *of* V *is given by*

$$\kappa(G, (A, R)) = \frac{\sum_{\{u,v\} \in C_{(A,R)}} |w(\{u, v\})|}{|E|}$$

Notice that there may not exist a unique partition with a maximum coherence value. Actually, at least two partitions have the same coherence value, since $\kappa(G, (A, R)) = \kappa(G, (R, A))$ for any partition (A, R) of V.

4 Conceptual Coherence in Description Logics

Thagard characterises conceptual coherence with these principles [19]:

Symmetry: Conceptual coherence is a symmetric relation between pairs of concepts.

Association: A concept coheres with another concept if they are positively associated, i.e., if there are objects to which they both apply.

Given Concepts: The applicability of a concept to an object may be given perceptually or by some other reliable source.

Negative Association: A concept incoheres with another concept if they are negatively associated, i.e., if an object falling under one concept tends not to fall under the other concept.

Acceptance: The applicability of a concept to an object depends on the applicability of other concepts.

To provide a precise account of these principles we shall formalise *Association* and *Negative Association* between concepts expressed in a description logic, since these are the principles defining coherence and incoherence. We shall assume coherence between two concept descriptions when we have explicitly stated that one subsumes the other ("there are objects to which both apply"); and we shall assume incoherence when we have explicitly stated that they are disjoint ("an object falling under one concept tends not to fall under the other concept").

Definition 4. *Given a Tbox $\mathcal{T}$ in description logic $\mathcal{AL}$ and a pair of concept descriptions $C, D \notin \{\top, \bot\}$, we will say that:*

- *C coheres with D, if $C \sqsubseteq D \in \mathcal{T}$, and that*
- *C incoheres with D, if $C \sqsubseteq \neg D \in \mathcal{T}$ or $C \sqcap D \sqsubseteq \bot \in \mathcal{T}$.*

In addition, coherence and incoherence between concept descriptions depend on the concept constructors used, and we will say that, for all atomic concepts A, atomic roles R, and concept descriptions $C, D \notin \{\top, \bot\}$:

- *$\neg A$ incoheres with A;*
- *$C \sqcap D$ coheres both with C and with D;*
- *$\forall R.C$ coheres (or incoheres) with $\forall R.D$, if C coheres (or incoheres) with D.*[1]

Symmetry follows from the definition above, and *Acceptance* is captured by the aim of maximising coherence in a coherence graph. For this we need to define how a TBox determines a coherence graph, and, in order to keep the graph finite, we express coherence and incoherence only between non-trivial concept descriptions (i.e., excluding $\top$ and $\bot$) that are explicitly stated in the TBox.

Definition 5. *Let $\mathcal{T}$ be a TBox in $\mathcal{AL}$. The set of* non-trivial subconcepts *of $\mathcal{T}$ is given as*

$$\mathit{sub}(\mathcal{T}) = \bigcup_{C \sqsubseteq D \in \mathcal{T}} \mathit{sub}(C) \cup \mathit{sub}(D)$$

where sub *is defined over the structure of concept descriptions as follows:*

$$\begin{aligned}
\mathit{sub}(A) &= \{A\} \\
\mathit{sub}(\bot) &= \emptyset \\
\mathit{sub}(\top) &= \emptyset \\
\mathit{sub}(\neg A) &= \{\neg A, A\} \\
\mathit{sub}(C \sqcap D) &= \{C \sqcap D\} \cup \mathit{sub}(C) \cup \mathit{sub}(D) \\
\mathit{sub}(\forall R.C) &= \{\forall R.C\} \cup \mathit{sub}(C) \\
\mathit{sub}(\exists R.\top) &= \{\exists R.\top\}
\end{aligned}$$

[1] Note that since $\mathcal{AL}$ allows only for limited existential quantification we cannot provide a general rule for coherence between concept descriptions of the form $\exists R.\top$.

Definition 6. *The* coherence graph of a TBox $\mathcal{T}$ *is the edge-weighted, undirected graph* $G = \langle V, E, w \rangle$ *whose vertices are non-trivial subconcepts of* $\mathcal{T}$ *(i.e.,* $V = \mathsf{sub}(\mathcal{T})$*), whose edges link subconcepts that either cohere or incohere according to Definition 4, and whose edge-weight function* w *is given as follows:*

$$w(\{C, D\}) = \begin{cases} 1 & \text{if } C \text{ and } D \text{ cohere} \\ -1 & \text{if } C \text{ and } D \text{ incohere} \end{cases}$$

5 Conceptual Blending in $\mathcal{AL}$

We follow the modelling principles and techniques of [4], where the process of conceptual blending is characterised by the notion of amalgams [1, 14]. According to this approach, the process of conceptual blending can be described as follows:

1. We take a taxonomy of concepts described in a background ontology expressed as a Tbox $\mathcal{T}$.
2. A mental space of an atomic concept A is modelled, for the purpose of conceptual blending, by means of a subsumption $A \sqsubseteq C$ specifying the necessary conditions we are focusing on.
3. The new concept to be invented is represented by the concept description that conjoins the atomic concepts to be blended.
4. With amalgams we generalise the input spaces based on the taxonomy in our TBox until a satisfactory blend is generated.

Formally, the notion of amalgams can be defined in any representation language $\mathcal{L}$ for which a subsumption relation between formulas (or descriptions) of $\mathcal{L}$ can be defined, and therefore also in the set of all $\mathcal{AL}$ concept descriptions with the subsumption relation $\sqsubseteq_{\mathcal{T}}$.

To formally specify an amalgam we first need to introduce some notions. Let N_C be a set of concept names, N_R be a set of role names, and $\mathcal{L}(\mathcal{T})$ be the finite set of all $\mathcal{AL}$ concept descriptions that can be formed with the concept and role names occurring in an $\mathcal{AL}$ TBox $\mathcal{T}$. Then:

Definition 7. *Given two descriptions* $C_1, C_2 \in \mathcal{L}(\mathcal{T})$*:*

- *A* most general specialisation *(MGS) is a description* C_{mgs} *such that* $C_{mgs} \sqsubseteq_{\mathcal{T}} C_1$ *and* $C_{mgs} \sqsubseteq_{\mathcal{T}} C_2$ *and for any other description* D *such that* $D \sqsubseteq_{\mathcal{T}} C_1$ *and* $D \sqsubseteq_{\mathcal{T}} C_2$*, then* $D \sqsubseteq_{\mathcal{T}} C_{mgs}$*.*
- *A* least general generalisation *(LGG) is a description* C_{lgg} *such that* $C_1 \sqsubseteq_{\mathcal{T}} C_{lgg}$ *and* $C_2 \sqsubseteq_{\mathcal{T}} C_{lgg}$ *and for any other description* D *such that* $C_1 \sqsubseteq_{\mathcal{T}} D$ *and* $C_2 \sqsubseteq_{\mathcal{T}} D$*, then* $C_{lgg} \sqsubseteq_{\mathcal{T}} D$*.*

Intuitively, an MGS is a description that has some of the information from both original descriptions C_1 and C_2, while an LGG contains what is common to them.

An *amalgam* or *blend* of two descriptions is a new description that contains *parts from these original descriptions* and it can be formally defined as follows.

Definition 8 (Amalgam). *Let $\mathcal{T}$ be an $\mathcal{AL}$ TBox. A description $C_{am} \in \mathcal{L}(\mathcal{T})$ is an* amalgam *of two descriptions C_1 and C_2 (with LGG C_{lgg}) if there exist two descriptions $\overline{C_1}$ and $\overline{C_2}$ such that: $C_1 \sqsubseteq_\mathcal{T} \overline{C_1} \sqsubseteq_\mathcal{T} C_{lgg}$, $C_2 \sqsubseteq_\mathcal{T} \overline{C_2} \sqsubseteq_\mathcal{T} C_{lgg}$, and C_{am} is an MGS of $\overline{C_1}$ and $\overline{C_2}$.*

The number of blends that satisfies the above definition can be very large and selection criteria for filtering and ordering them are therefore needed. Fauconnier and Turner discussed optimality principles [7], however, these principles are difficult to capture in a computational way, and other selection strategies need to be explored. Since we use a logical theory such as $\mathcal{AL}$, one way to evaluate a blend is consistency checking. Another alternative, that we will investigate in this paper, is to evaluate blends in terms of conceptual coherence.

The LGG and the generalised descriptions, needed to compute the amalgam as defined above, are obtained by means of a generalisation refinement operator that allows us to find generalisations of $\mathcal{AL}$ concept descriptions.

5.1 Generalising $\mathcal{AL}$ descriptions

Roughly speaking, a generalisation operator takes a concept C as input and returns a set of descriptions that are more general than C by taking a Tbox $\mathcal{T}$ into account.

In order to define a generalisation refinement operator for $\mathcal{AL}$, we define the upward cover set of atomic concepts. In the following definition, $\mathsf{sub}(\mathcal{T})$ (Definition 5) guarantees the following upward cover set to be finite.

Definition 9. *Let $\mathcal{T}$ be an $\mathcal{AL}$ TBox with concept names from N_C. The* upward cover set *of an atomic concept $A \in N_C \cup \{\top, \bot\}$ with respect to $\mathcal{T}$ is given as:*

$$\begin{aligned} \mathsf{UpCov}(A) := \{C \in \mathsf{sub}(\mathcal{T}) \cup \{\top, \bot\} \mid A \sqsubseteq_\mathcal{T} C \\ \text{and there is no } C' \in \mathsf{sub}(\mathcal{T}) \cup \{\top, \bot\} \\ \text{such that } A \sqsubset_\mathcal{T} C' \sqsubset_\mathcal{T} C\} \end{aligned} \tag{1}$$

We can now define our generalisation refinement operator for $\mathcal{AL}$ as follows.

Definition 10. *Let $\mathcal{T}$ be an $\mathcal{AL}$ TBox. We define the* generalisation refinement operator γ *inductively over the structure of concept descriptions as follows:*

$$\begin{aligned} \gamma(A) &= \mathsf{UpCov}(A) \\ \gamma(\top) &= \mathsf{UpCov}(\top) = \emptyset \\ \gamma(\bot) &= \mathsf{UpCov}(\bot) \\ \gamma(C \sqcap D) &= \{C' \sqcap D \mid C' \in \gamma(C)\} \cup \{C \sqcap D' \mid D' \in \gamma(D)\} \cup \{C, D\} \\ \gamma(\forall r.C) &= \begin{cases} \{\forall r.C' \mid C' \in \gamma(C)\} & \text{whenever } \gamma(C) \neq \emptyset \\ \{\top\} & \text{otherwise.} \end{cases} \\ \gamma(\exists r.\top) &= \emptyset \end{aligned}$$

$\mathsf{House} \sqsubseteq \mathsf{Object}$	$\mathsf{Resident} \sqsubseteq \mathsf{Person}$
$\mathsf{Boat} \sqsubseteq \mathsf{Object}$	$\mathsf{Passenger} \sqsubseteq \mathsf{Person}$
$\mathsf{Land} \sqsubseteq \mathsf{Medium}$	$\mathsf{Person} \sqcap \mathsf{Medium} \sqsubseteq \bot$
$\mathsf{Water} \sqsubseteq \mathsf{Medium}$	$\mathsf{Object} \sqcap \mathsf{Medium} \sqsubseteq \bot$
$\mathsf{Water} \sqcap \mathsf{Land} \sqsubseteq \bot$	$\mathsf{Object} \sqcap \mathsf{Person} \sqsubseteq \bot$

Fig. 1. The background ontology of the House and Boat.

We should notice at this point that γ can return concept descriptions that are equivalent to the concept being generalised. One possible way to avoid this situation is to discard these generalisations [4]. Given a generalisation refinement operator γ, $\mathcal{AL}$ concepts are related by refinement paths as described next.

Definition 11. *A finite sequence $C_1, \ldots, C_n$ of $\mathcal{AL}$ concepts is a* concept refinement path $C_1 \xrightarrow{\gamma} C_n$ *from C_1 to C_n of the generalisation refinement operator γ iff $C_{i+1} \in \gamma(C_i)$ for all $i : 1 \leq i < n$. $\gamma^*(C)$ denotes the set of all concepts that can be reached from C by means of γ in a finite number of steps.*

The repetitive application of the generalisation refinement operator allows us to find a description that represents the properties that two or more $\mathcal{AL}$ concepts have in common. This description is a common generalisation of $\mathcal{AL}$ concepts, the so-called *generic space* that is used in conceptual blending.

Definition 12. *An $\mathcal{AL}$ concept description G is a generic space of the $\mathcal{AL}$ concept descriptions $C_1, \ldots, C_n$ if and only if $G \in \gamma'^*(C_i)$ for all $i = 1, \ldots, n$.*

5.2 An Example: The House-Boat Blend

The process of conceptual blending in terms of amalgams can be illustrated by means of a typical blend example: the *house-boat* [7, 8]. The precise formalisation is not critique at this point, different ones exist [9, 15], but all provide similar distinctions.

The $\mathcal{AL}$ theories for House and Boat introduce the axioms modelling the mental spaces for *house* and *boat*.

$$\mathsf{House} \sqsubseteq \forall \mathsf{usedBy}.\mathsf{Resident} \sqcap \forall \mathsf{on}.\mathsf{Land}$$
$$\mathsf{Boat} \sqsubseteq \forall \mathsf{usedBy}.\mathsf{Passenger} \sqcap \forall \mathsf{on}.\mathsf{Water}$$

The House and Boat theories cannot be directly blended since they generate an inconsistency. This is due to the background ontology stating that the medium on which an object is situated cannot be *land* and *water* at the same time (Figure 1). Therefore, some parts of the House and Boat descriptions need to be generalised in a controlled manner before these concepts can be blended. The generic space between a house and a boat—an object that is on a *medium* and *used-by* a *person*—is a lower bound in the space of generalisations that need to be explored in order to generalise these concepts and to blend them into a *house-boat*. The generic space is obtained according to Definition 12 by applying the refinement operator γ.

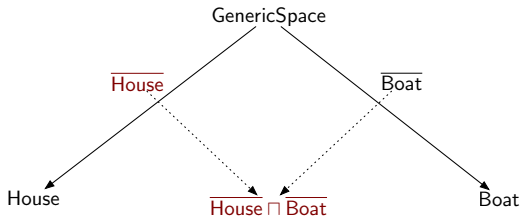


Fig. 2. A diagram of an amalgam HouseBoat from descriptions House and Boat and their respective generalisations $\overline{\text{House}}$ and $\overline{\text{Boat}}$. Arrows indicate the subsumption of the target by the source of the arrow.

Example 1. Let us consider the House and Boat concepts. Their generic space is: ∀usedBy.Person⊓ ∀on.Medium and is obtained as follows. In the House concept, the subconcepts ∀usedBy.Resident and ∀on.Land are generalised to ∀usedBy.Person and ∀on.Medium respectively. In the Boat concept, the subconcepts ∀usedBy. Passenger and ∀on.Water are generalised in a similar way.

From a conceptual blending point of view, the *house-boat* blend can be created when the medium on which a house is situated (land) becomes the medium on which boat is situated (water), and the resident of the house becomes the passenger of the boat. This blend can be obtained when the input concepts house and boat are generalised as follows:

$$\overline{\text{House}} \sqsubseteq \forall \text{usedBy.Resident} \sqcap \forall \text{on.Medium}$$
$$\overline{\text{Boat}} \sqsubseteq \forall \text{usedBy.Person} \sqcap \forall \text{on.Water}$$

The *house-boat* blend is obtained by conjoining the generalised mental spaces $\overline{\text{House}}$ and $\overline{\text{Boat}}$ (Figure 2). It is easy to see that $\overline{\text{House}} \sqcap \overline{\text{Boat}}$ is an amalgam according to Definition 8.

6 Evaluating the Coherence of Conceptual Blends

This section describes how coherence is used to evaluate blends. That is, how coherence graphs are built, and how the different coherence values are to be interpreted. The overall idea is to compute the coherence graph and maximsing partitions for each blend, and use the maximal coherence degree of the coherence graphs to rank the blends.

Let $\mathcal{T}$ be the TBox of the background ontology, let $A \sqsubseteq C$ and $B \sqsubseteq D$ be the axioms representing our mental spaces, and let $A \sqcap B$ be the new concept we would like to invent. The process of evaluating blends according to conceptual coherence can be described as follows:

1. Given the mental spaces, we generate a candidate blend according to Definition 8.

2. We form the coherence graph for $\mathcal{T} \cup \{A \sqsubseteq C, B \sqsubseteq D\}$, including node $A \sqcap B$, according to Definition 6.
3. We compute the coherence maximising partitions according to Definition 3 and we associate it to the blend.
4. We repeat this procedure for all the blends that can be generated from the mental spaces.

Once the maximising partitions are computed, the coherence of the blend could be measured in terms of the coherence value of the coherence-maximising partitions. The degree of the coherence graph directly measures how much a blend coheres with the background ontology.

Definition 13. *Let $G = \langle V, E, w \rangle$ the coherence graph of a blend B and let $\mathcal{P}$ the set of partitions of G. The maximal coherence value of B of G is* $\deg(B) = \max_{P \in \mathcal{P}} \{\kappa(G, P)\}$.

This maximal coherence value can be used to rank blends as follows.

Definition 14. *Let $\mathcal{T}$ be a TBox of a background ontology, let $A \sqsubseteq C$ and $B \sqsubseteq D$ be the axioms representing mental spaces, let $\mathcal{B}$ be the set of blends that can be generated from them. For each $b_1, b_2 \in \mathcal{B}$, we say that b_1 is preferred to b_2 ($b_1 \succeq b_2$) if and only if* $\deg(b_1) \geq \deg(b_2)$.

To exemplify how the coherence degree can be used to evaluate blends, we consider the *house-boat* example. According to the amalgams process of conceptual blending described in the previous section, several blends can be generated by blending the mental space of House and Boat. In particular, the concept $\mathsf{House} \sqcap \mathsf{Boat}$ is a valid blend.

The coherence graph blending the House and Boat directly is shown in Figure 3. As expected the concepts House and Boat positively coheres with the axioms representing the mental spaces and with the concept $\mathsf{House} \sqcap \mathsf{Boat}$, which is representing the blend. The incoherence relation between $\forall \mathsf{on.Land}$ and $\forall \mathsf{on.Water}$ is due to the fact that the concepts Water and Land incohere, since the background ontology contains the disjointness axiom $\mathsf{Water} \sqcap \mathsf{Land} \sqsubseteq \bot$. The coherence graph of House and Boat has a maximal coherence value of 0.84.

For the sake of our example, we generate new blends by generalising the axioms modelling our mental spaces. For instance, by applying the generalisations seen in the previous section that lead to the creation of the *house-boat* blend, we obtain the coherence graph in Figure 4.[2] The coherence graph of blending $\overline{\mathsf{House}}$ and $\overline{\mathsf{Boat}}$ has a maximal coherence value of 0.9. This graph yields a higher coherence degree since generalising $\forall \mathsf{on.Land}$ to $\forall \mathsf{on.Medium}$ prevents the appearance of the incoherence relation between $\forall \mathsf{on.Land}$ and $\forall \mathsf{on.Water}$.

It is easy to see that the blend $\overline{\mathsf{House}} \sqcap \overline{\mathsf{Boat}}$ is preferred to $\mathsf{House} \sqcap \mathsf{Boat}$ since it has a maximal coherence degree that is higher.

[2] Concepts belonging to the background ontology are omitted.

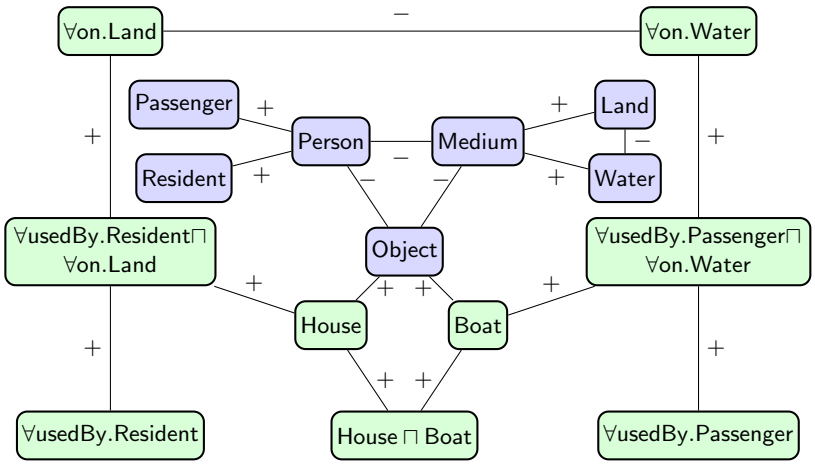


Fig. 3. The coherence graph of the House ⊓ Boat blend, showing the main concepts and their coherence relations. Blue and green coloured boxes represent concepts belonging to the background ontology and to the input mental spaces respectively.

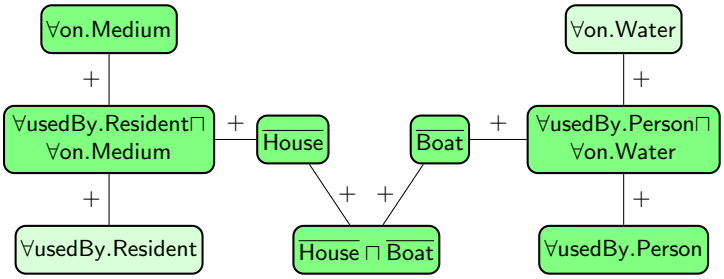


Fig. 4. The coherence graph of the $\overline{\text{House}} \sqcap \overline{\text{Boat}}$ blend, showing the main concepts and coherence relations. Generalised concepts are displayed in a darker tonality.

7 Conclusion

This paper should be seen as a first attempt to (a) provide a formal account of conceptual coherence for a particular concept representation language, and (b) to explore its applicability for guiding the process of conceptual blending.

With respect to (a), we proposed a formalisation of conceptual coherence between concept descriptions expressed in the basic $\mathcal{AL}$ description logic. This is only a starting point, and obviously this formalisation exercise should be carried out for more expressive concept representation languages. Usually, coherence and incoherence are not treated only in binary terms, but it is also natural to take certain degrees of coherence or incoherence into account. This, for instance, has also been the approach of Joseph et al. when formalising deductive coherence [11]. Although there is not an obvious way to do so with the formalisation of

conceptual coherence of $\mathcal{AL}$ proposed in this paper, we do not discard that this could be done for more expressive concept representation languages. One could imagine that description logics with number restrictions or nominals, such as $\mathcal{SROIQ}$ for instance, would allow for expressing degrees of concept overlap that could be interpreted as degrees of coherence or incoherence.

With respect to (b), we have so far only focused on how the coherence values of a graph of concept descriptions were evolving dependent on how these descriptions were changing in our amalgam-based conceptual blending process. However, we have not discussed yet an other important aspect of coherence theory, namely how to interpret the two parts of a coherence-maximising partition: the set of accepted and of rejected concepts. The information that a particular concept description falls in the set of accepted concepts or in the set of rejected concepts could also be taken into account to decide the acceptance or rejection of newly invented concepts; or even of already existing concepts in the background knowledge, in the light of newly invented concepts. With the formalisation in $\mathcal{AL}$ given in this paper we could not see yet a clear way to provide such an interpretation of acceptance and rejection, but we think this aspect might become clearer as a wider range of concept representation languages is explored.

In this paper we attempted to see how coherence could be used as another tool for guiding the process of conceptual blending and for evaluating conceptual blends in the task of concept invention; an additional technique to those already proposed, such as optimality principles [16], logical consistency [13], and values of audiences [3]. We believe it is worth to further study the proper combination of these techniques and to carry out a comprehensive evaluation.

An implementation of conceptual coherence presented in this paper using the OWL API and Answer Set Programming is available at: `https://rconfalonieri@bitbucket.org/rconfalonieri/coinvent-coherence.git`.

References

[1] T. R. Besold and E. Plaza. Generalize and Blend: Concept Blending Based on Generalization, Analogy, and Amalgams. In *Proceedings of the 6th International Conference on Computational Creativity, ICCC15*, 2015.

[2] M. A. Boden. *The Creative Mind: Myths and Mechanisms*. George Weidenfeld and Nicolson Ltd., 1990.

[3] R. Confalonieri, E. Plaza, and M. Schorlemmer. A Process Model for Concept Invention. In *Proc. of the 7th International Conference on Computational Creativity, ICCC16*, 2016.

[4] R. Confalonieri, M. Schorlemmer, O. Kutz, R. Peñaloza, E. Plaza, and M. Eppe. Conceptual blending in EL++. In *Proc. of 29th Int. Workshop on Description Logics*, volume 1577 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016.

[5] M. Eppe, R. Confalonieri, E. Maclean, M. A. Kaliakatsos-Papakostas, E. Cambouropoulos, W. M. Schorlemmer, M. Codescu, and K. Kühnberger. Computational Invention of Cadences and Chord Progressions by Conceptual Chord-Blending. In *Proc. of the 24th Int. Joint Conf. on Artificial Intelligence, IJCAI 2015*, pages 2445–2451. AAAI Press, 2015.

[6] G. Fauconnier and M. Turner. Conceptual integration networks. *Cognitive Science*, 22(2):133–187, 1998.

[7] G. Fauconnier and M. Turner. *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities*. Basic Books, New York, 2003.

[8] J. Goguen. An introduction to algebraic semiotics, with application to user interface design. In *Computation for Metaphors, Analogy, and Agents*, volume 1562 of *Lecture Notes in Computer Science*, pages 242–291. Springer, 1999.

[9)] J. A. Goguen and D. F. Harrell. Style: A computational and conceptual blending-based approach. In S. Argamon, K. Burns, and S. Dubnov, editors, *The Structure of Style*, chapter 12, pages 291–316. Springer, 2010.

[10] M. Guhe, A. Pease, A. Smaill, M. Martínez, M. Schmidt, H. Gust, K.-U. Kühnberger, and U. Krumnack. A computational account of conceptual blending in basic mathematics. *Cognitive Systems Research*, 12(3–4):249–265, 2011.

[11] S. Joseph, C. Sierra, M. Schorlemmer, and P. Dellunde. Deductive coherence and norm adoption. *Logic Journal of the IGPL*, 18(1):118–156, 2010.

[12] Z. Kunda and P. Thagard. Forming unpressions from stereotypes, traits, and behaviours: A parallel-constraint-satisfaction theory. *Psychological Review*, 103 (2):284–308, 1996.

[13] F. Neuhaus, O. Kutz, M. Codescu, and T. Mossakowski. Fabricating monsters is hard. towards the automation of conceptual blending. In *Proceedings of the Workshop "Computational Creativity, Concept Invention, and General Intelligence" 2014*, volume 01-2014 of *Publications of the Institute of Cognitive Science*, 2014

[14] S. Ontañón and E. Plaza. Amalgams: A Formal Approach for Combining Multiple Case Solutions. In *Proceedings of the International Conference on Case Base Reasoning*, volume 6176 of *Lecture Notes in Computer Science*, pages 257–271. Springer, 2010.

[15] F. C. Pereira. *Creativity and Artificial Intelligence: A Conceptual Blending Approach*. Mouton de Gruyter, 2007a.

[16] F. C. Pereira and A. Cardoso. Optimality principles for conceptual blending: A first computational approach. *AISB Journal*, 1(4), 2003.

[17] B. Ran and P. R. Duimering. Conceptual combination: Models, theories and controversies. *International Journal of Cognitive Linguistics*, 1(1):65–90, 2010.

[18] P. Thagard. Coherent and creative conceptual combinations. In *Creative thought: An investigation of conceptual structures and processes*, pages 129–141. American Psychological Association, 1997.

[19] P. Thagard. *Coherence in thought and action*. The MIT Press, 2000.

[20] M. Turner. Blending and conceptual integration. http://markturner.org/ blending.html. Last checked on June 20, 2016.

[21] T. Veale and D. O'Donoghue. Computation and blending. *Cognitive Linguistics*, 11(3/4):253–281, 2000.

An Argument-based Creative Assistant for Harmonic Blending
Paper type: System Description Paper

Maximos Kaliakatsos-Papakostas[a], Roberto Confalonieri[b], Joseph Corneli[c], Asterios Zacharakis[a] and Emilios Cambouropoulos[a]

[a]Department of Music, Aristotle University of Thessaloniki, Greece
[b]Artificial Intelligence Research Institute, IIIA-CSIC, Bellaterra (Barcelona), Spain
[c]Department of Computing, Goldsmiths, University of London, UK
[a]{max,aszachar,emilios}@mus.auth.gr
[b]confalonieri@iiia.csic.es
[c]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., 2015; 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).

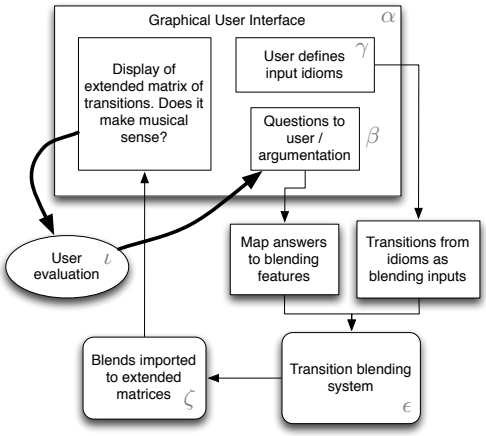


Figure 1: A schematic diagram of the system's workflow.

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 blending, 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 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).

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 higher level arguments as inputs to the system is equivalent to allowing a musical expert to interact with it in a language he/she understands. This offers the user a flexible way to adjust the harmonic blending properties according to different input scenarios in order to improve the creativity of the system. Extended experimentation with the system —by making use of the available arguments— can enable music experts to provide valuable feedback regarding the functionality of transition features, thus directly intervening in the blending process by answering simple questions. In a future scenario, the assessment of the system will be based on merely musicological criteria that should be more clearly defined.

The paper is organised as follow. In the next section, we describe the harmonic blending creative process embedded in a creative assistant tool we implemented. Next, we describe the methodology of transition blending and extending harmonic spaces. We show how user arguments are used to evaluate transition blends based on two criteria resembling two of the optimality principles of conceptual blending. Then, we present a process-based system evaluation that focuses on the creative acts of programmers (Colton et al., 2014). This evaluation is helpful in guiding further developments of the system. These are discussed in a concluding section.

System overview and test cases

Figure 1 illustrates a diagram of the presented system. The user (music expert) interacts with the system through the Graphical User Interface (GUI), where she/he selects two initial idioms (harmonic spaces) in γ and defines the important features used in conceptual blending by answering to specific questions (argumentation) in β. The selected initial idioms are described as sets of chord transitions, while the provided answers to questions are mapped to enabling/disabling features of transitions (see Section 'Chord transitions description and blending') that define the outcome of transition blending (see Section 'Evaluation of transition blending via arguments').

Afterwards, pairs of transition in the two initial harmonic spaces are given as inputs to the transition blending system in ϵ where new transitions are invented through conceptual blending. These transitions are then integrated into an *extended* musical idiom that includes the initial idioms selected by the user, while the role of the new transitions is to provide musically meaningful connections between the initial harmonic spaces. The created extended idiom is displayed to the user in the GUI in terms of a transition matrix (see Section 'From transition blends to transition matrices'). By observing the matrix, the music expert evaluates (ι) the results produced by the current blending setup, i.e., the given questions to the argumentation module (β), and re-adjusts her/his answers in β accordingly.

Several scenarios for initial idiom combinations are available to the user. The system included several harmonic blending test cases according to which the user could blend simple 'artificial' harmonic spaces as well as harmonic spaces trained from data in different tonalities. The artificial harmonic spaces are constructed to include simple transitions in order to allow clear interpretations of the results, e.g., a C major space included the chords C, F and G7. Among the trained idioms that have been examined, there are sets of Bach chorales in major and minor mode, and sets of modal chorales in several modes.

The test cases, in which harmonic spaces in different tonalities are blended, resemble the musical task of finding transition paths for tonality modulations (changing the tonality of a given harmonic space). This task allowed music experts to identify arguments for defining the features of transition blending that connect potentially remote harmonic spaces (e.g., C major with F♯ major) in a manner that is explainable in music theory in terms of tonality modulations. Through the processes offered by the system, the music experts were able to come to conclusions about what transition features are important for constructing meaningful connections between different combinations of pairs of initial harmonic spaces.

Methodological aspects of transition blending and extending harmonic spaces

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[1] (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., 2015b). The amalgam-based workflow generalises input concepts until a generic space is found and

[1] http://www.coinvent-project.eu

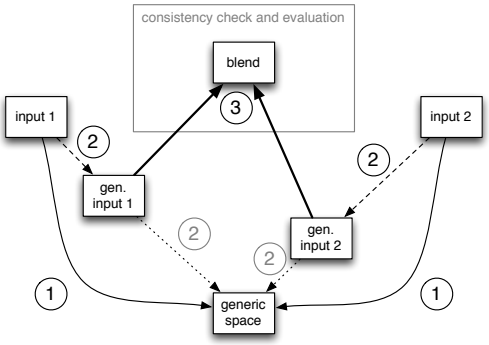


Figure 2: Conceptual blending based on amalgam. The generic space is computed (1) and the input spaces are successively generalised (2), while new blends are constantly created (3). Some blends might be inconsistent or purely evaluated according to blending optimality principles or domain specific criteria.

'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).[2]

Amalgam-based conceptual blending has been applied to invent chord cadences (Eppe et al., 2015a; 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 I_1: G7 $\rightarrow$ C and I_2: B$\flat$m $\rightarrow$ 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 I_1: G7 $\rightarrow$ C and I_2: B$\flat$m $\rightarrow$ 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 I_1 and the I_2 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 I_2) and tries to unify the generalised versions of the inputs again.

[2]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.

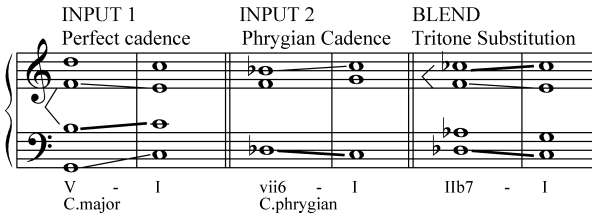


Figure 3: Example of blending cadences, which are special case of transitions, where blending the perfect and the Phrygian produce the tritone substitution cadence blend.

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.

Blending completion (Fauconnier and Turner, 2002) is a domain-specific process that uses background knowledge to consistently assign specific properties to generalised terms. In the hitherto discussed example, blend completion is used for completing the $A\flat$ note as the chord's fifth in blending the perfect and Phrygian cadence in order to obtain the tritone substitution cadence (Figure 3).

After several blends have been computed, an 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[3]. 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, violating the symmetry principle:* Each of the input spaces is a trivial form of a blend. This is a bad blend, because no information from the other input spaces is considered.
- *Example 2, violating the web principle:* Consider a blend that includes all properties of the generic space, but all other information are filled by properties that are not included in any of the input spaces. This blend has the least possible connections with the input spaces and therefore, the least amount of information from the inputs is identifiable in this blend.

These examples suggest two criteria for ranking the blends; we provide a computational characterisation of them below.

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

[3]The amalgam-based algorithm produces many blends by following alternative generalisation paths.

input transitions for blending, producing a list of blended transitions that are afterwards embeded 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, Katsiavalos, 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. *fromRoot*: the root pitch class of the first chord,
2. *toRoot*: the root pitch class of the second chord,
3. *fromType*: the type of the first chord (GCT base),
4. *toType*: the type of the second chord (GCT base),
5. *fromPCs*: the pitch classes included in the first chord,
6. *toPCs*: the pitch classes included in the second chord,
7. *DICinfo*: the DIC vector of the transition,
8. *DIChas0*: Boolean value indicating whether the DIC of the transition has 0,
9. *DIChas1*: Boolean value indicating whether the DIC of the transition has 1,
10. *DIChasMinus1*: Boolean value indicating whether the DIC of the transition has -1,
11. *ascSemZero*: Boolean value indicating whether the first chord has the relative pitch class value 11,
12. *descSemZero*: Boolean value indicating whether the first chord has the relative pitch class value 1,
13. *semZero*: Boolean value indicating whether the first chord has the relative pitch class value 11 or 1,
14. *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. *descSemNextRoot*: Boolean value indicating whether the first chord has a pitch class with descending semitone relation with the pitch class of the second chord's root,
16. *semNextRoot*: Boolean value indicating whether the first chord has a pitch class with ascending or descending semitone relation with the pitch class of the second chord's root, and
17. *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 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. First-order Markov models, specifically, indicate the probability of transitions from one chord to another, disregarding information about previous chords. Therefore, 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 (this neutralises the diagonal). 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?*

Among many possible answers, the idea examined in the present system is to create an *extended* transition matrix that includes not only an altered version of the initial ones, but also 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 $\mathtt{I_i}$, only chords of the i-th idiom are used. When using the transition probabilities in $\mathtt{I_i}$, the resulting harmonisations preserve the character of idiom i.
2. Transitions in $\mathtt{A_{i-j}}$ create direct jumps from chords of the i-th to chords of the j-th idiom. If a blended transition happens to be in $\mathtt{A_{i-j}}$ there is no need for further considerations – such a transition can be included in the extended matrix.

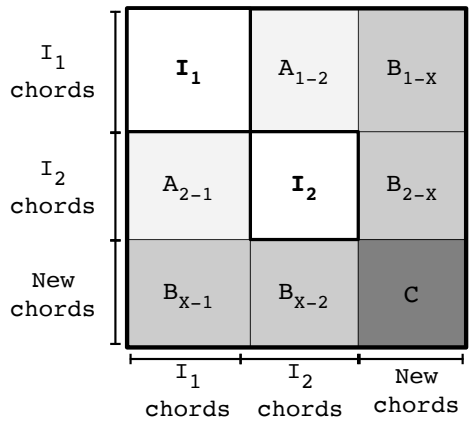


Figure 4: Graphical description of an *extended* matrix that includes transition probabilities of both initial idioms and of several new transitions generated through transition blending. These new transitions allow moving across the initial idioms, creating a new extended idiom that is a superset of the initial ones.

3. Transitions in $\mathtt{B}_{\mathtt{i-X}}$ constitute harmonic motions from a chord of idiom i to a newly created chord by blending. Similarly, transitions in the $\mathtt{B}_{\mathtt{X-j}}$ 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: $c_i \rightarrow c_x$ followed by a transition $c_x \rightarrow c_j$, 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 $\mathtt{B}_{\mathtt{i-X-j}}$.
4. Sector C transitions incorporate pairs of chords that exist outside the i-th and j-th idioms. Having two external chords, transitions in C violate our hypothesis for moving from one known chord sets to the other using one new chord at most; therefore, blends resulting to C-type blends are disregarded.

Based on this analysis of the extended matrix, a methodology is proposed for using blends between transitions in $\mathtt{I}_1$ and $\mathtt{I}_2$. Thereby, transitions in $\mathtt{I}_1$ are blended with ones in $\mathtt{I}_2$ and a number of the 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 ($\mathcal{B}$), 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 $\mathtt{A}_{\mathtt{i-j}}$ or in $\mathtt{B}_{\mathtt{i-X-j}}$.

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 idioms, 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.

Question	Chord Properties	Transition Changes
Q1	*fromRoot* *toRoot* *fromType* *toType*	
Q2	*fromRelPCs* *toRelPCs*	
Q3		*DIChas0*
Q4		*DIChas1* *DIChasN1*
Q5		*DIChas2* *DIChasN2*
Q6		*DICinfo*
Q7		*ascSemZero* *descSemZero* *semZero*
Q8		*ascSemNextRoot* *descSemNextRoot* *semNextRoot*
Q9		*5thRootRel*

Table 1: Abstraction of chords' and transition changes' features.

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).

Q1: Are roots and types of chords important?

Q2: Are individual pitch classes of chords important?

Q3: Are repeating pitch classes in transitions important?

Q4: Are semitone steps in transitions important?

Q5: Are tone steps in transitions important?

Q6: Are the intervalic contents of transitions important?

Q7: Are semitone motions to the tonic important?

Q8: Are semitones to the second chord's root important?

Q9: Are motions of the chord roots by 5^{th} important?

The first two questions concern characteristics of the chords that constitute the transition, mapping the user answers to features from 1 to 6, while the remaining seven questions concern intervalic changes that occur within the transition, mapping the user answers to features from 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 $\mathcal{Q}$. 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 $\langle q, F\rangle$, where $q \in \mathcal{Q}$ and $F \subset \mathcal{F}$.*

The user can specify at most 9 arguments, each of them is mapped to a set of properties. The set of user arguments $\{A_1, \ldots, A_9\}$ corresponding to answers to $\mathcal{Q}$ will be denoted by $\mathcal{A}$. We assume to have a function $\psi : \mathcal{A} \rightarrow \mathcal{F}$ 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 $\mathsf{a}(b, I)$, is defined as:

$$\mathsf{a}(b, I) = \sum_{A_i \in \mathcal{A}} \mathsf{Val}(A_i, b, I)$$

where $\mathsf{Val} : \mathcal{A} \rightarrow \mathbb{R}$ 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, Val is defined in different ways.

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

$$\mathsf{Val}(A_1, b, I) = \sum_{F_j \in \psi(A_1)} \mathsf{equals}(F_j(I), F_j(b))$$

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), Val is defined as:

$$\mathsf{Val}(A_2, b, I) = \sum_{F_j \in \psi(A_2)} |F_j(I) \cap F_j(b)|$$

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 intervalic contents of transitions (A_6), Val is defined as:

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

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, Val is defined as:

$$\mathsf{Val}(A_i, b, I) = \sum_{f_j \in \psi(A_i)} 1 - (f_j(I) - f_j(b))$$

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

$$\mathsf{assoc}(b) = \sum_{I_i \in \mathcal{I}} \mathsf{a}(b, I_i)$$

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 $\mathsf{asym}(b)$, is calculated as:

$$\left| \frac{\mathsf{a}(b, I_1)^2 + \mathsf{a}(b, I_1)\mathsf{a}(b, I_2)}{\mathsf{a}(b, I_1)^2 + \mathsf{a}(b, I_2)} - \frac{\mathsf{a}(b, I_2)^2 + \mathsf{a}(b, I_1)\mathsf{a}(b, I_2)}{\mathsf{a}(b, I_2)^2 + \mathsf{a}(b, I_1)} \right|$$

The value of $\mathsf{asym}(b)$ 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.

$$\mathsf{rate}(b) = \frac{\mathsf{assoc(b)}(1 - \mathsf{asym}(b))}{\mathsf{assoc(b)} + (1 - \mathsf{asym}(b))}$$

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* $\mathsf{rate}(b_1) \geq \mathsf{rate}(b_2)$.

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 α, the user has access to modules γ, and β 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 ϵ embodies the (process-level) concept of a system that make use of the supplied idioms and the blending properties to generate *example* transition

matrices, ζ. In the current version of the system, these transitions are evaluated by the user (music expert) in step ι using sophisticated harmonic knowledge that reflects historically established musical *aesthetic*. The user can then return to the GUI α, and adjust the settings of γ and β to regenerate the transitions.

This is illustrated in Figure 5 in box **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 $\overline{C_p^\epsilon}$ 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 $\overline{C_g^\gamma}$, $\overline{C_g^\beta}$ 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 $\overline{A_g^\iota}$.

The *system's* primary responsibilities take place through the creative acts E_g^ϵ, which generate blends, and $S[a_g^{\beta,\gamma}](e^{\epsilon *})$, in which the aesthetic $A_g^{\beta,\gamma}$ (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, $S[a_g^{\beta,\gamma}](e^{\epsilon *})$ 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 behavior. This as-yet hypothetical situation is illustrated in the box **P2**.

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

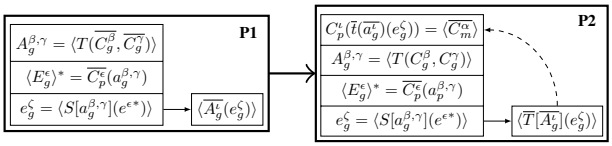


Figure 5: The current implementation **P1** prototypes automated evaluation of blends according to user's arguments; this points to a proposed future implementation **P2** with further automation.

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 readjust 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.

References

Amgoud, L., and Prade, H. 2009. Using arguments for making and explaining decisions. *Artificial Intelligence* 173(3-4):413–436.

Antovic, M. 2009. Musical Metaphors in Serbian and Romani Children: An Empirical Study. *Metaphor and Symbol* 24(3):184–202.

Antovic, M. 2011. Musical metaphor revisited: Primitives, universals and conceptual blending. *Universals and Conceptual Blending (February 17, 2011)*.

Bench-Capon, T. J. M., and Dunne, P. E. 2007. Argumentation in artificial intelligence. *Artificial Intelligence* 171(10-15):619–641.

Bonet, B., and Geffner, H. 1996. Arguing for decisions: A qualitative model of decision making. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence (UAI'96)*, 98–105.

Cambouropoulos, E.; Kaliakatsos-Papakostas, M.; and Tsougras, C. 2014. An idiom-independent representation of chords for computational music analysis and generation. In *Proceeding of the joint 11th Sound and Music Computing Conference (SMC) and 40th International Computer Music Conference (ICMC)*, ICMC–SMC 2014.

Cambouropoulos, E.; Katsiavalos, A.; and Tsougras, C. 2013. Idiom-independent harmonic pattern recognition based on a novel chord transition representation. In *In Proceedings of the 3rd International Workshop on Folk Music Analysis (FMA2013)*, FMA 2013.

Cambouropoulos, E. 2012. A Directional Interval Class Representation of Chord Transitions. In *Proc. of the 12th International Conference for Music Perception and Cognition, & 8th Conference of the European Society for the Cognitive Sciences of Music)*, ICMPC-ESCOM 2012.

Colton, S.; Pease, A.; Corneli, J.; Cook, M.; and Llano, T. 2014. Assessing progress in building autonomously creative systems. In Ventura, D.; Colton, S.; Lavrac, N.; and Cook, M., eds., *Proceedings of the Fifth International Conference on Computational Creativity*.

Confalonieri, R.; Schorlemmer, M.; Plaza, E.; Eppe, M.; Kutz, O.; and Peñaloza, R. 2015. Upward Refinement for Conceptual Blending in Description Logic – An ASP-based Approach and Case Study in EL^{++}. In *International Workshop on Ontologies and Logic Programming for Query Answering, International Joint Conference on Artificial Intelligence (IJCAI) 2015*.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77(2):321 – 357.

Eppe, M.; Confalonieri, R.; Maclean, E.; Kaliakatsos-Papakostas, M.; Cambouropoulos, E.; Schorlemmer, M.; Codescu, M.; and Kühnberger, K.-U. 2015a. Computational invention of cadences and chord progressions by conceptual chord-blending. In *International Joint Conference on Artificial Intelligence (IJCAI) 2015*.

Eppe, M.; Maclean, E.; Confalonieri, R.; Kutz, O.; Schorlemmer, M.; and Plaza, E. 2015b. ASP, Amalgamation, and the Conceptual Blending Workflow. In Calimeri, F.; Ianni, G.; and Truszczynski, M., eds., *Logic Programming and Nonmonotonic Reasoning*, volume 9345 of *LNCS*. Springer International Publishing. 309–316.

Fauconnier, G., and Turner, M. 2002. *The Way We Think: Conceptual Blending And The Mind's Hidden Complexities*. Basic Books.

Goguen, J., and Harrell, D. F. 2010. Style: A Computational and Conceptual Blending-Based Approach. In Argamon, S., and Dubnov, S., eds., *The Structure of Style: Algorithmic Approaches to Understanding Manner and Meaning*. Berlin: Springer. 147–170.

Goguen, J. 2006. Mathematical Models of Cognitive Space and Time. In Andler, D.; Ogawa, Y.; Okada, M.; and Watanabe, S., eds., *Reasoning and Cognition*, volume 2 of *Interdisciplinary Conference Series on Reasoning Studies*. Keio University Press.

Kaliakatsos-Papakostas, M., and Cambouropoulos, E. 2014. Probabilistic harmonisation with fixed intermediate chord constraints. In *Proceeding of the joint 11th Sound and Music Computing Conference (SMC) and 40th International Computer Music Conference (ICMC)*, ICMC–SMC 2014.

Kaliakatsos-Papakostas, M.; Zacharakis, A.; Tsougras, C.; and Cambouropoulos, E. 2015. Evaluating the General Chord Type representation in tonal music and organising GCT chord labels in functional chord categories. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR 2015)*.

Ontañón, S., and Plaza, E. 2010. Amalgams: A formal approach for combining multiple case solutions. In *Proc. of the Int. Conf. on Case Base Reasoning*, volume 6176 of *Lecture Notes in Computer Science*, 257–271. Springer.

Ritchie, G. D. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds and Machines* 17:67–99.

Schorlemmer, M.; Smaill, A.; Kühnberger, K.-U.; Kutz, O.; Colton, S.; Cambouropoulos, E.; and Pease, A. 2014. Coinvent: Towards a computational concept invention theory. In *5th Int. Conf. on Computational Creativity*.

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

Zacharakis, A.; Kaliakatsos-Papakostas, M.; and Cambouropoulos, E. 2015. Conceptual blending in music cadences: A formal model and subjective evaluation. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR 2015)*.

Zbikowski, L. M. 2002. *Conceptualizing Music: Cognitive Structure, Theory, and Analysis*. Oxford University Press.