Interpretation-driven mapping: a framework for conducting search and re-representation in parallel for computational analogy in design

Submitted to:
AIEDAM Special Issue on Analogical Thinking

On:
July 16, 2014

Authors:
Kazjon Grace[1], John Gero[1,2] and Rob Saunders[3]

Affiliations:
[1] College of Computing and Informatics, UNC Charlotte.
[2] Krasnow Institute for Advanced Study, George Mason University.
[3] Faculty of Architecture, Design and Planning, The University of Sydney.

Contact Details (first author):
Email: k.grace@uncc.edu
Phone: 347 443 6271
Mail: 255 W. MLK Jr. Blvd, Apt #505, Charlotte 28202, NC.

Pages: 39
Figures: 8
Tables: 6

Short Title: Interpretation-driven mapping

Interpretation-driven mapping: a framework for conducting search and re-representation in parallel to support computational analogy in design

Abstract

This paper presents a framework for the interactions between the processes of mapping and re-representation within analogy-making. Developing analogical reasoning systems for use in design tasks requires representations that are fluid and open to reinterpreting. The framework, Interpretation Driven Mapping, casts the process of constructing an analogical relationship as requiring iterative, parallel interactions between mapping and representation. This paper argues that this interpretation-driven approach focusses research on a fundamental problem in analogy-making: how do the representations that make new mappings possible emerge during the mapping process? The framework is useful for both describing existing analogy-making models and designing future ones.

The paper presents the development of a computational model informed by the framework, Idiom, that learns ways to reinterpret the representations of objects as it attempts to map between them. In order to demonstrate its feasibility, the results of an implementation of the Idiom model in the domain of visual analogy are presented. Several analogies constructed by the system are presented as examples.

The Interpretation Driven Mapping framework is then used to compare representational change in Idiom and three previously published systems. Symbolic, hybrid and connectionist models of analogical mapping differ widely in the ways they represent and solve analogical problems, but the framework is used to elicit the similarities between the ways their representations change. By comparing these three families of analogy-making systems in this way this paper demonstrates that Idiom combines the large scope of possible re-representations observed in connectionist approaches with the context-specific guidance observed in hybrid and symbolic approaches.

Keywords

Computational analogy-making, representation, interpretation.

1 Introduction

The production of a new analogy involves the construction of a new, often complex, relationship between two objects that was not previously part of the system's knowledge (Gick and Holyoak, 1980; Gentner, 1983; Holyoak, 1996; Hummel and Holyoak, 1997; Hofstadter, 2008). This new relationship, which we call an "association" is then used to transfer knowledge from the source domain to that of the target (Gick and Holyoak, 1983; Detterman and Sternberg, 1993; Barnden and Holyoak, 1994; Holyoak et al., 1994; Gentner and Holyoak, 1997; Robertson, 2000). We concern ourselves with the former process – mapping – which is integral to both analogical reasoning and related cognitive processes such as conceptual blending (Fauconnier and Turner, 2003), metaphor (Lakoff and Johnson, 2003), bisociation (Koestler, 1967), simile (Veale and Hao, 2007) allegory (Fletcher, 1964), and inference (Wang, 2009).

Terminological precision in this field often suffers from the ability of many relevant words to refer both to processes and their results, so we adopt the following definitions throughout this paper. *Mapping* is the process of searching a source and a target object for a set of shared relationships on which to base a new analogy, while an *association* is the resulting set of relationships. *Interpreting* is the process of changing the representation of source and target objects to enable mapping, while a *transformation* is the resulting function which produces the new object representations from the prior ones. The processes of mapping and interpreting operate in parallel, iteratively generating *candidate* associations and transformations until a satisfactory pair of candidates – an association and the transformation which enables it – is found. Following Gentner's Structure Mapping (Gentner, 1983) we adopt the notion that an association connects a set of relationships between features of one object's representation to an analogous set of relationships between another object's representation. Analogy depends on compatible relational representations (Doumas et al., 2008; Penn et al., 2008; Holyoak, 2012). Our focus is on how that representation can be re-interpreted *during and because of* the process of analogical mapping, and the effect of this reinterpreting on the resultant association.

Viewed with this focus, the process of mapping can be considered a search of the space of possible new relationships between two objects, where a new relationship is defined as a previously undiscovered commonality. This commonality can be a shared feature, a shared relationship between components, or any other matching representational structure. Due to the combinatorial nature of a space of possible mappings the search can be computationally expensive, but conceptually the problem is simple: find matching pairs of representational elements. Many models of analogy that perform this kind of search exist, with ANALOGY (Evans, 1964) being a very early example and the Structure Mapping Engine (Falkenhainer et al., 1989) being a well-known one. These models, and the numerous computational implementations of them, demonstrate

that locating such shared properties between objects is a solved problem *when compatible representations already exist.*

The representation of an object that is useful in a particular analogy-making context may not be the representation that is most familiar to the observer. Contextually salient features may not be typically regarded as central to an object's description, and nor might be the representational structure in which those features are placed. Consider the case of an architect seeking inspiration from nature in the design of a stadium in a hot climate. She considers as a source the ladybug, with its hard outer shell (the *elytra*) covering foldable wings, and constructs an analogy to the stadium's retractible roof. A design concept is synthesised with an outward-opening roof segment that reveals a foldable shade cloth underneath. This example of analogy-by-design demonstrates that analogical mapping does not necessarily concern the design attributes that are considered most salient to an object considered in isolation (such as a wing's ability to grant flight). It is extremely unlikely that any agent's default representation of a ladybug considers the transparency of its wings and the path taken by the elytra upon opening. Two potential approaches to constructing this analogy exist: either all attributes are available to the agent in all possible representational structures all the time, or representations are constructed in response to the emerging analogical context, interpreted and re-interpreted as mapping proceeds. This research develops a model of analogy-making based on notion of contextual representation: the idea that the hemispherical shape of a ladybug's wing-covers became salient only in the context of the form of a stadium.

For an analogy-making model to be an effective component of any computational design reasoning system it must take account of the representational fluidity and temporally entwined problem framing/problem solving nature of the design process. To this end we have developed the "Interpretation-Driven Mapping" framework, which describes the parts of the analogy-making process that focus on the ways representations change during the search. Exploration in computer-aided design has been defined as transforming the search space, rather than searching within it(Gero, 1994). Escaping from what was previously thought of as the space of the possible has also been proposed as a fundamental component of creativity (Boden, 1998). Using this approach, we develop a framework intended to force new transformations of the object representations and permit them to influence mapping, and derive both a computational model and a working implementation from it. This enables our structure-mapping-based model to construct analogies that would not be possible without this ability to reinterpet. We evaluate our proof-of-concept implementation (and the model from which it derives) by this capability. Our model is able to construct associations that would not have been possible in other SMT-based models without specifically crafted representations. As we are forcing new associations that previously would not have been possible, then this capability serves as a proof-of-concept of the approach.

We apply this framework in the construction of a computational model of analogical mapping, Idiom, and describe several examples generated by an implementation of that model that could not have been constructed via structure-mapping without interpretation-driven mapping. The implementation of Idiom is intended to serve as a proof-of-concept of both the framework and model, demonstrating their computational feasibility and potential as an approach to analogy-making. We then compare the way representations change in Idiom and three other analogy-making systems, demonstrating the differences of our approach.

1.1 Representational transformation in analogy-making

The driver for a focus on re-representation during mapping is applications of analogy-making in the computational modelling of design reasoning. Analogical reasoning has been identified as a core component in the synthesis of new design elements (Goel, 1997) and in creative cognition as a whole (Thagard, 1997; Dunbar, 2001). Increased use of analogical reasoning in design tasks has been associated with both designer expertise and design quality (Casakin and Goldschmidt, 1999). Design is an iteratively reflective (Schön, 1983) and a wicked, ill-defined (Rittel, 1987) problem, characterised by representational structures that are highly mutable. The problem of making computational analogies in the real world has been called "messy" because of the dynamic, situated nature of real-world concepts (Rissland, 2006). Design concerns not only concepts that exist, but hypothetical concepts yet to exist. Creative design has been characterised as occurring only when the set of design variables that comprise the solution changes during the process, not just their values or ranges of possible values (Gero, 1990). Analogy has also been observed in use at multiple levels of the design process, applying to the design problem at hand directly as well as the process being used to solve it (Visser, 1996; Goel, 1997).

The focus of this research is on how the two objects with initially disparate representations can be reinterpreted in such a way as to be compatible for mapping. This parallels the distinction between *constructing* analogies and *solving* them identified in Harpaz-Itay et al. (2006), with the former being characterised by improved transferability in human subjects. Previous approaches to this problem have focussed on building representations from sub-symbolic elements (Hofstadter and Mitchell, 1992), iteratively constructing relational representations (Doumas et al., 2008), reconstructing representations from memory into the current context (Kokinov and Petrov, 2001), or by activating progressively more abstract concepts (Petkov et al., 2011). Models of analogy-making in design have focussed on projecting representations into an ontology that characterises their function(s), and then mapping based on patterns (Visser, 1996; Qian and Gero, 1996; Griffith et al., 1996; Bhatta and Goel, 1997; Griffith et al., 2000), a special case of mapping by abstraction. This work seeks to define a generally applicable framework for how representations change during the

process of mapping that can both inform future computational models and serve to compare these existing approaches to representational transformation.

2 The *interpretation-driven mapping* framework

Constructing representations suitable for analogy in a complex, creative domain such as design is a complex task. At the heart of this complexity is a chicken-and-egg question: an association cannot be found without a representation that supports it, and yet whether a representation contains mappable elements cannot be determined without searching. Assuming the unavailability of representations that express the relations encapsulated within both objects in a compatible fashion, neither representation nor mapping can completely precede the other and they must take place interactively and in parallel. We develop a framework for conceptualising analogy which centres on this parallel, iterative re-interpreting. The central premise of our framework is that the current state of mapping should inform future representations, and the current state of representation should inform further mapping.

Given symbolic representations of a source and target object, we model how those representations change over the course of the mapping process. We focus on this component of the analogy-making process to investigate how the context of searching for a mapping between source and target affects their representations. To facilitate this focus we consider representation change *during* mapping separately from representation construction *prior to* mapping. The latter has long been considered a central component of analogy-making and incorporated into models (Koestler, 1967; Wolstencroft, 1989; Chalmers et al., 1992; French, 2002), we focus on the former as a distinct but equally important consideration. This focus on representational change during mapping echoes the discussion on ontological mismatches in (Davies et al., 2003).

We propose a framework, "Interpretation-driven Mapping" (IDM), for describing the mapping process that incorporates this notion of representational transformation. In this formulation (and using *interpreting is dependent on candidate associations*, i.e. how representations change depends on the progress of the search for mappings, and *mapping is dependent on candidate transformations*, i.e. how mappings are searched for is dependent on the progress of the search for representational changes. Our framework's name derives from this interaction: mapping is interpretation-driven and vice versa. As a consequence of our desire for generality in this framework we do not place any constraints on how the objects are represented, how their symbolic representations are initially constructed, what constitutes a satisfactory pair of candidates, or what happens to the association/transformation pair once it has been found.

Prior approaches to re-representation during analogical mapping include the *minimal ascension* method for extending Structure Mapping in Falkenhainer (1990), in which non-identical relationships can be consid-

ered contextually mappable only when they have a common superordinate in the hierarchy of relationships. This is similar to the ontological approach used in Qian and Gero (1996), where relationships are mappable if they perform the same function, and to that used in McDermott (1979) and Davies et al. (2003), where attributes are abstracted iteratively until rendered mappable. Such approaches can be extended to leverage relational hierarchies from external sources like WordNet, as in Holyoak and Thagard (1989), or from hierarchies inferred from large text corpora, as in Turney (2008). Our iterative strategy for re-representing and mapping is similar to that adopted in Yan et al. (2003), although the mechanisms for interpreting in IDM are more general than those proposed in that work.

The IDM framework, Figure 1, highlights the interactions between the re-representation of objects and the search for mappings. The perception process(es) that are precursors to the interpreting/mapping cycle, as well as the transfer process(es) that follow it are included for illustrative purposes only: IDM makes no assumptions about those processes. We acknowledge transfer's crucial role in analogy-making (Winston, 1978; Gick and Holyoak, 1983; Hall, 1989; French, 2002), and its absence is a matter of scope, not of exclusion. IDM's focus is the iterative and cyclical relationship between mapping and interpreting.

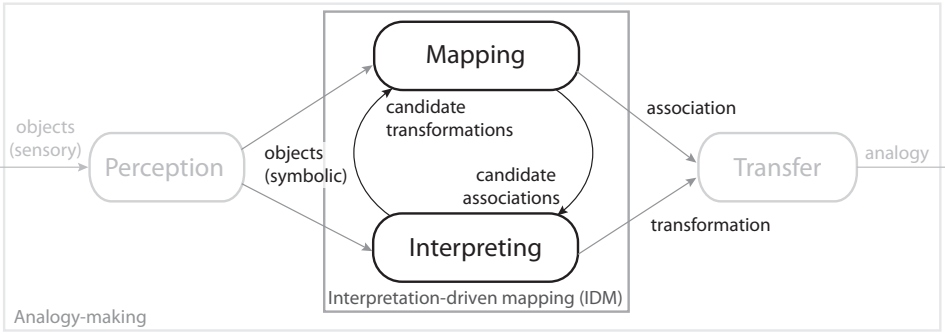

Figure 1: The Interpretation-driven mapping (IDM) framework (centre) embedded in a model of computational analogy. IDM addresses representation construction and parallelised search for mappings and object re-interpreting.

It has been argued (Chalmers et al., 1992; Kokinov and Petrov, 2001; French, 2002) that the perception of analogues-to-be is not a precursor to the search for mappings but a concurrent process. The IDM framework supports and extends this notion, with representational change during mapping being its principal driver. While the "Perception" process in Figure 1 operates prior to the processes of the framework, this sequential and discretised depiction allows us to focus on how representations *change* during the mapping process. We separate the processes acting to produce the representation of an object that exists when mapping begins ("Perception") from the processes that act to change the representation of that object during mapping ("Interpreting") based on their roles only, and make no assumptions about the similarity or difference in

their mechanisms. We also refrain from stating that the symbolic representation which is present at the start of mapping is in any way "untransformed" or "canonical", based on literature in embodied, constructivist and situated cognition (Clancey, 1997; Schacter et al., 2000; Mahon and Caramazza, 2008) that disputes the existence of such Platonic conceptualisations. The initial representation is merely the representation existing at the time mapping begins, and the basis for representational transformations applied during that process.

The IDM framework does not prescribe analogy-making completely, it is a framework for the interactions between representation and mapping, not a framework for the whole of computational analogy. The framework also does not make any assumptions about the nature of the mapping and interpreting processes or the representation of objects, transformations, and associations. This distinguishes it from previous approaches, such as the VAE model (Sowa and Majumdar, 2003), which utilise a similar structure but limit possible transformations. The framework addresses the "which-came-first?" question in representation in analogy-making by positing that the new relationship between objects and the representations that enable it arise out of a simultaneous, interactive process. Using the principles of IDM it becomes possible to describe how any computational analogy-making model incorporates representational change into mapping. While other models were not derived from our framework, framing analogical representation in this way can enable scrutinisation and comparison of otherwise disparate systems. The three core principles of the framework described, and the questions that arise from them when inspecting an analogy-making model through the lens of IDM, are as follows:

- *Mapping can be conceptualised as a search for an association between source and target.* Constructing a new association (a set of shared relationships) is a search through a space of possible associations. Mapping is an iterative process that produces and evaluates candidate associations. From this principle two questions arise when viewing a model through the lens of IDM:
  1. What bounds the space of valid associations constructible by mapping?
  2. What causes an association to be selected by mapping?

- *Interpreting can be conceptualised as search for a transformation of source and target.* Constructing representations that will be used in the association produced by mapping is a search through a space of possible transformations (to be applied to the initial object representations). interpreting is an iterative process that produces and evaluates candidate transformations. From this principle two questions arise when viewing a model through the lens of IDM:
  3. What bounds the space of valid transformations constructible by interpreting?
  4. What causes a transformation to be selected by interpreting?

- *Interpreting occurs in parallel with mapping.* The two processes operate simultaneously, with candidate transformations affecting mapping and candidate associations affecting interpreting. From this principle two questions arise when viewing a model through the lens of IDM:
    5. How do candidate transformations selected by interpreting affect mapping?
    6. How do candidate associations selected by mapping affect interpreting?

3 Idiom: a computational model derived from the IDM framework

Idiom is a computational model of the association-construction component of analogy-making based on the IDM framework. Idiom demonstrates the feasibility of the the principles of parallel search and re-representation, and is so named for its ability to construct and operate on non-literal representations of objects. The model describes one way representations can change during the construction of an analogy, and how those changes both affect and are affected by the search for mappings. Idiom provides one possible set of answers for the six questions raised at the end of Section 2, although they are by no means the only answers possible.

Interpretation-driven mapping involves two bi-directionally interactive processes; mapping (or more generally the search for solutions) and interpreting (or more generally the re-framing of the problem). The Idiom model adds a third process: Perception, by which the representations comprised of object features and the relationships between them are constructed from the sensory representations initially observed by the system. As an instantiation of the IDM framework Idiom outputs a new association (a set of shared relationships between the source and target objects) and the representational transformation under which that association exists.

3.1 Model overview

We develop a symbolic representation of the three processes which comprise Idiom – *perception, interpreting and mapping.* A high level view of Idiom, Figure 2, shows the symbols associated with each of the processes.

Our model of representational change in analogy-making begins with the model observing sensory representations of two objects. We do not treat the "source" and "target" objects differently in this model as in the portion of the process we are modelling the two are interchangeable. We refer these objects as "sensory representations" to emphasise that there is no veridical representation of any analogue, only observations made through the system's perceptual faculties. The symbols used to represent the Idiom model are as

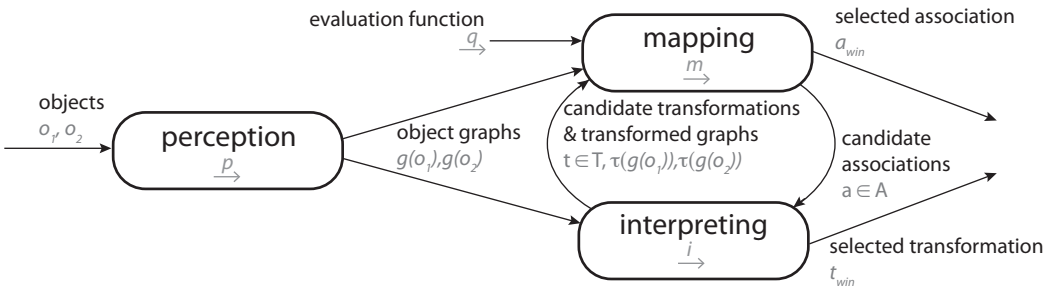

Figure 2: The three processes of Idiom, showing the symbolic representations of their inputs and outputs. Idiom is a computational model that instantiates the IDM framework

follows:

a	is a candidate association. An association $a \in A$ is an ordered nonempty list of pairs of features, with the first of each pair being a node in $\tau(g(o_1))$ and the second being a node in $\tau(g(o_2))$.
a_{win}	is a candidate association that has been evaluated as satisfactory under the evaluation function $\stackrel{q}{\rightarrow}$, leading to the Idiom model ceasing its search.
A	is the set of candidate associations being considered by the mapping process.
$g(o_1), g(o_2)$	are the graph representations of the two objects that result from perception.
o_1, o_2	are the sensory representations of the two objects.
t	is a candidate transformation.
t_{win}	is the transformation that was applied when a satisfactory association was discovered.
T	is the set of candidate transformations being considered by the interpreting process.
τ	is a candidate transformation that has been applied to the graph representations.
$\tau(g(o_1)), \tau(g(o_2))$	are the graph representations after the active transformation has been applied.
$\stackrel{i}{\rightarrow}$	is the process of interpreting that uses the current association candidates to generate new ways to transform the graph representations, and then selects one transformation to apply to the graphs. See Section 3.4 for details.
$\stackrel{m}{\rightarrow}$	is the process of mapping that searches the interpreted graph representations for associations based on Structure Mapping Theory. See Section 3.3 for details.
$\stackrel{p}{\rightarrow}$	is the process of perception that constructs graph representations comprised of features and relationships between them from the sensory representations. See Section 3.2 for details.
$\stackrel{q}{\rightarrow}$	is an evaluation function which can be applied to associations.

The model's perception process, $\stackrel{p}{\rightarrow}$ constructs the graph representations used in search, and can be represented as (1). Graph representations of objects in which nodes are features and edges are relationships are a commonly adopted representation in analogy-making (Gentner, 1983; Holyoak and Thagard, 1989; Hofstadter and Mitchell, 1992; Kokinov and Petrov, 2001) although by no means the only possible representation. We adopt them in Idiom to demonstrate one possible route to implementing representational change, although the IDM framework makes no such representational assumption.

$$\overset{p}{\rightarrow} := o_1 \rightarrow g(o_1), o_2 \rightarrow g(o_2) \tag{1}$$

The model's mapping process, $\overset{m}{\rightarrow}$ searches for associations between graph representations, and can be represented as (2). As input it takes the transformed graph objects for searching and the set of candidate associations for updating. In Idiom the Mapping process is dependent only on the current active transformation and its affects on the graph representations. This is a simplification of the IDM framework, in which mapping is reliant on the set of candidate transformations, allowing multiple transformations to influence mapping simultaneously, rather than a winner-takes-all approach used here. We denote the set of candidate associations after an iteration of the mapping process as A' to indicate it has been updated. Idiom follows the Structure Mapping Theory (Gentner, 1983) definition of analogical mapping, in which a mapping exists between a common subset of relationships between the features of the objects, although we reserve the word "mapping" for the process. As in Section 2 we refer to the shared subset of relationships between features that results from mapping as an "association". We embed Gentner's theory in a system where the graph representations are iteratively transformed by an interpreting process in parallel with the search for mappings.

$$\overset{m}{\rightarrow} := \tau(g(o_1)), \tau(g(o_2)), A \rightarrow a_{win} \vee A' \tag{2}$$

The model's interpreting process, $\overset{i}{\rightarrow}$, uses the set of candidate associations to construct new ways of transforming the graphs and selects one of them to apply, and can be represented as (3). As input it takes the object graphs and the set of candidate associations for use in generating new transformations, and both the current active transformation and the set of candidate transformations for searching and evaluation. We denote the active transformation and set of transformations after an iteration of the interpreting process as τ' and T' respectively to indicate they have been updated. Transformations act on the graph representations, and may include the addition or removal of nodes, or the addition, removal or relabelling of edges. New transformations are generated based on what changes could improve the performance of associations in A according to the same success measure used to select candidate mappings, $\overset{q}{\rightarrow}$. Transformations are generated, evaluated and stored, with the most promising transformation at each point in time becoming the active transformation τ that is applied to the graphs. This characterisation of analogical mapping as being transformation-dependent echoes the definition of similarity as "representational distortion" (Hahn et al., 2003; Hodgetts et al., 2009).

$$\overset{i}{\rightarrow} := g(o_1), g(o_2), A, \tau, T \rightarrow T', \tau', \tau'(g(o_1)), \tau'(g(o_2)) \tag{3}$$

Associations can be evaluated by $\xrightarrow{q}$ under any transformation, allowing the system to iteratively improve its candidate associations based on the available transformations and its candidate transformations based on the available associations. A candidate association may specify erroneously mapped feature pairs (those that form graphs not connected by any shared relationships within their respective objects, or those containing nodes that do not exist under the current active transformation). These invalid shared associations are ignored by $\xrightarrow{q}$.

Mapping and interpreting both operate incrementally, with each incrementally updated set of candidate associations A being the basis for an updated τ, which is then in turn the basis for an updated A and so on. This iterative interaction implements the fundamental principles of the IDM framework. Below we describe the three processes in more detail.

3.2 Perception

The perception process in the Idiom model, $\xrightarrow{p}$, takes low-level sensory representations of a form appropriate for the domain in which it is implemented and constructs from them the graphs of features and relationships used in mapping. To construct graph representations $\xrightarrow{p}$ extracts features from the sensory representations, clusters those features into concepts, constructs relationships between features based on their sensory and conceptual contexts, and then compiles the features and their relationships into graphs. Details of the perception processes are particularly affected by the choice of an implementation domain, and in this model specification we adopt an abstract framing to maintain broad applicability. The six processes that make up perception, Figure 3, are:

- *Feature detection* identifies and describes elements of the sensory representation which become the features of the object on which analogies will be based. Feature definitions are domain specific.
- *Concept recognition* classifies features into known conceptual categories based on a domain specific similarity function.
- *Concept generation* constructs new conceptual categories from the features that were not able to be classified. The set of known concepts is updated with these new categories.
- *Typological relationship construction* describes relations between features based on the concepts that they instantiate, for example between two features that are instances of similar concepts.
- *Topological relationship construction* describes relationships between features using spatial information from the sensory representations. The space in question may be literal (as in a spatial relationship

like "above" or "beside") or it may be a conceptual space (Boden, 2003; Gärdenfors, 2004) in which spatial relationships have a metaphorical meaning.

- *Graph construction* translates the relational information into appropriately formatted edge labels, converting, rounding and grouping relations into labels and expresses the representation as a graph for each object where features are nodes and relations are edges.

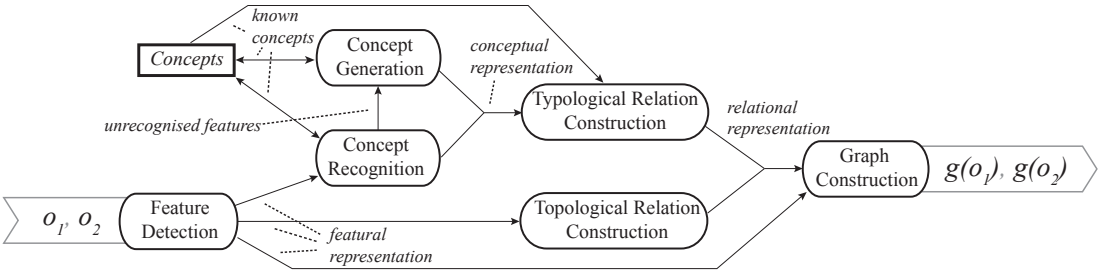

Figure 3: The structure of the perception process of our model. The sensory representations are converted into featural, conceptual, relational and finally graph representations.

Feature detection acts on the sensory representations o_1 and o_2 to extract a set of features. A "feature" is a notion specific to the domain in which Idiom is implemented, and may be any element of the object, either a discrete component or a description of some aspect of the whole. For example, in a musical domain a feature might be a phrase of music, while in a visual domain a feature might be a shape. These features are not present in o_1 and o_2, which are of a more primitive nature, such as the sequence of samples making up an audio file in the case of the former example or the matrix of pixels making up an image in the case of the latter.

Each feature has a description embodying its properties, which is represented as a list of attribute/value pairs the nature of which is domain dependent. The model requires that it be possible to calculate the similarity between two features for the purposes of categorisation, but otherwise makes no assumption of the nature of features or their representations. Features are the nodes in the graph representations used by Idiom, and the remainder of the perception process concerns the construction of the relationships between them that make up the edges in those graphs.

Concept recognition is one of the two processes involved in clustering the feature representations into conceptual groups for the purpose of inferring typological relationships between them. Concept recognition attempts to place the features extracted by feature detection into known categories, which we refer to as "concepts". A concept is a region of the space of possible features, and the concept recognition process determines which of the features observed in the objects belong to existing concepts. The set of known concepts

persists over multiple iterations of the Idiom association-construction process. Features not determined to be an instance of an existing concept are passed to the concept generation process.

Concept generation is the second of the processes involved in classifying features by their conceptual categories, and occurs after the concept recognition process has completed. Those features which were not classifiable within the current conceptual hierarchy are placed into new concepts as needed. The first time the Idiom model runs this by necessity occurs with all concepts. After both concept recognition and concept generation have operated on the feature representations all features will have been tagged as members of at least one conceptual category. The concept-tagged feature representations are then used in the construction of relationships between features, with the conceptual information being used to determine typological relationships and the featural information being used to determine topological representations.

Typological relationship construction produces relationships between features based on the conceptual categories into which they have been placed. A relationship consists of an ordered pair of features and a relation that connects them. The Idiom model makes no assumptions about the structure of this relation other than that it expresses some property of the second feature in the pair, the 'destination feature' in relation to the first feature, the 'originating feature'. Typological relationships express relations between the concepts two features instantiate, for example "the originating feature instantiates the same concept as the destination feature", "the originating feature instantiates a similar concept as the destination feature", or "a concept instantiated by the originating feature's is a parent concept of one instantiated by the destination feature". These relations can have any data structure or content as determined by the nature of the relationship construction processes in an Idiom implementation. It is expected that in implementing the relationship construction processes of Idiom it would be necessary to specify a set of typological relationship types and the means by which they can be constructed.

Topological relationship construction produces relationships between features based on the structure of the sensory representations. They are identical in structure to typological relationships, consisting of an ordered pair of originating and destination feature along with the relation that connects them. However, the relationships between features described by topological relationship construction concern not the concepts instantiated by those features but the properties and context of those features within the sensory representation. Relations are expressed relative to the originating feature to increase their generality and enable mapping. For example, a topological relationship could express that "the destination feature is above the originating feature", "the destination feature is twice the size of the originating feature", or "the destination feature is a component of the originating feature". Whether these relations describe a physical topology, a compositional hierarchy or some other structure depends on the domain in which Idiom is implemented.

Graph construction compiles into a graph the features extracted by the feature detection process with the relationships from by the typological and topological construction processes. The features become the nodes in each object's graph and the relations that connected them become edges. The descriptions of the features are not directly used in mapping, and all information about them is encoded relatively through the edge labels. The resulting structures are directed (due to the originating/destination feature separation), labelled (due to the relations attached to each edge) multigraphs (in which two nodes can be connected by multiple edges expressing different relations). The two graph structures are then searched based on Structure Mapping Theory (Gentner, 1983): features in the source and target that share a pattern of relationships can become part of an association. In accordance with the IDM framework Idiom performs this search whilst iteratively reinterpreting the object graphs to enable otherwise-impossible associations.

3.3 Mapping

The mapping process in the Idiom model, $\xrightarrow{m}$, searches the graph representations provided by $\xrightarrow{p}$ in the context of the transformations applied by $\xrightarrow{i}$. Mapping and interpreting iteratively search and transform the graph representations until a suitable association is constructed. An individual iteration of the mapping process produces an updated set of candidate associations on which the next iteration of the interpreting process can operate. Mapping operates on the principles of Structure Mapping (Gentner, 1983), searching for sets of relationships between features *within* the individual objects that form shared patterns *between* the objects. For example, two objects may both have a pattern of features connected by a series of "immediately above" relationships, and an association could be constructed between those two patterns. The two processes that make up $\xrightarrow{m}$, Figure 4, are:

- *Association search* generates new candidate associations based on shared relationships between the transformed source and target graph representations. While there may not be a sufficient pattern of such relationships in the untransformed graphs, the transformation acts to facilitate better associations.
- *Association evaluation* evaluates the patterns of relationships found by association search using the quality measure $\xrightarrow{q}$. Each candidate receives an evaluation via that measure, with evaluations over a threshold resulting in an association being deemed suitable. If a suitable association is found the process returns a solution, completing this run of the Idiom model. If no suitable mapping is found the set of current candidates is returned for use by the interpreting system in potentially choosing a new active transformation.

Association search discovers associations – sets of pairs of features which can be considered analogous – by searching for shared relationships in the transformed graph representations. This is computationally

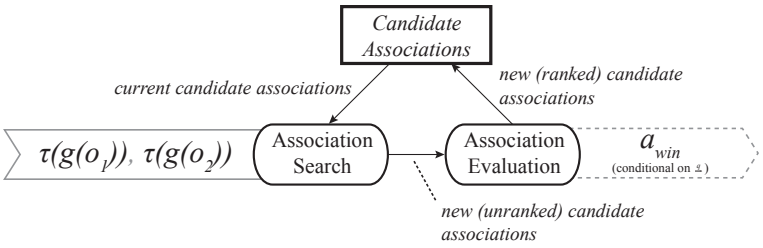

Figure 4: The structure of the mapping process of our model. The two transformed object graphs are searched for associations that are suitable under the evaluation measure.

modelled in Idiom as a search of the graph representations for edge-labelled isomorphic subgraphs[1]. This search applies not to the graph representations produced by perception, $g(o_1)$ & $g(o_2)$, but to those graphs reinterpreted in the context of the current active transformation τ. In the case of the very first iteration of the association search process τ defaults to a null transformation in which the graphs are unchanged, meaning that initially $g(o_1) \Leftrightarrow \tau(g(o_1))$ & $g(o_2) \Leftrightarrow \tau(g(o_2))$.

Association evaluation determines the value of each candidate association as measured by $\stackrel{q}{\rightarrow}$. While the specific measure is domain specific, Structure Mapping requires that these edge labels (which represent relationships between features of the source and target) must match exactly for them to be considered a valid component of the mapping. $\stackrel{q}{\rightarrow}$ can be implemented to evaluate mappings in a variety of ways, including measures based on the size (in nodes) of the mapped subgraph or measures based on the domain-dependent meaning of the features or relationships within that subgraph.

Graph representations within Idiom can contain multiple edges between the same two nodes. Associations which map a pair of nodes within the source to a pair of nodes within the target must map at least one of these edges to be valid, but need not match all of them. For example assume a group of features within one object are all the same size (expressed in topological relationships between them) and all instantiate the same concept (expressed in typological relationships between them). Within the second object is a group of features that all share the same size, but instantiate different concepts. The association evaluation process would find an association between these two groups to be valid based on the shared size relationships even though there were unmatched "same concept" relationships. This is the advantage of reducing complex relations to a set of simple relations each expressed on its own edge. The unmatched relationships could be associated in the context of an appropriate transformation, but not without.

The mapping process performs one iteration of the search for and evaluation of candidate associations and updates the set of candidates. If a candidate association a is discovered which scores sufficiently highly

[1]Complete maximum subgraph isomorphism search is a computationally intractable problem(Garey and Johnson, 1979; Kann, 1992), but here approximate solutions are acceptable.

at $\stackrel{q}{\rightarrow}$ then the Idiom model terminates, returning a and τ. Otherwise $\stackrel{i}{\rightarrow}$ is executed and the iterative cycle of searching and transforming the graph representations continues.

3.4 Interpreting

The interpreting process in the Idiom model, $\stackrel{i}{\rightarrow}$, uses the current set of association candidates produced by the last iteration of $\stackrel{m}{\rightarrow}$ to determine a new active transformation from among the candidates. To do this $\stackrel{i}{\rightarrow}$ constructs, evaluates and applies transformations. There is no requirement that the active transformation τ change with every iteration of the process, but it has the possibility to do so. The three processes that make up $\stackrel{i}{\rightarrow}$, Figure 5, are:

- *Transformation search* constructs new transformations by searching the current set of candidate associations for what could make them successful.
- *Transformation evaluation* considers all transformations in the current set of candidate transformations against the candidate associations to determine which should be the active transformation τ.
- *Transformation application* applies the newly updated active transformation τ' to the graph representations of the source and target objects, reinterpreting them and affecting the next iteration of the mapping process.

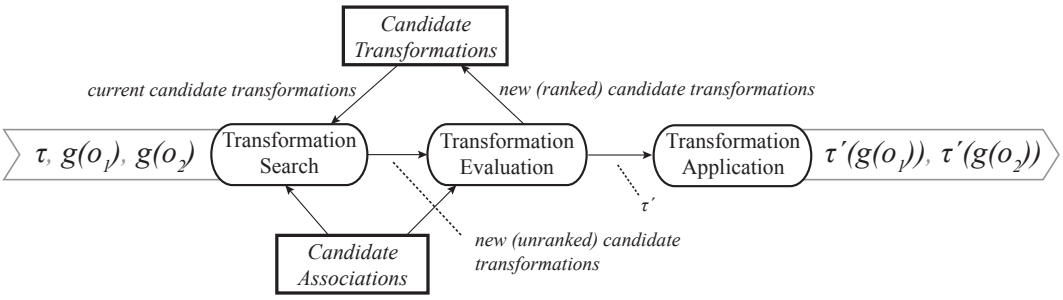

Figure 5: The structure of the interpreting process of the Idiom model. Transformations are generated from candidate associations and added to the current set of candidate transformations. From this set one is selected as the active transformation τ and applied to the graph representations.

Transformation search is the process by which new transformations used in interpreting are constructed. A generate-and-test approach is used to create transformations that would make associations in the current candidate set more successful at mapping between the two objects. Transformations are applied to the graphs being searched by $\stackrel{m}{\rightarrow}$. Each transformation must be general enough to be applied to any object graph possible within the domain(s) being used in analogy-making, although each transformation need not have

an effect on all graphs in the domains. This open-ended definition enables transformations to be constructed specific to the implementation of Idiom, but possibilities include the deletion, addition, consolidation or splitting of features, as well as the addition, deletion or relabelling of edges. Implementations of Idiom may govern these graph changes in any way, such as executing a modified version of the Perception process to produce altered representations. In this case the transformation would, in effect, be one of $\stackrel{p}{\rightarrow}$. The only requirement is that the result of the transformation be a transformed graph.

As the purpose of interpreting is to render non-matching relationships matching, the simplest transformation(s) involve the relabelling of graph edges. An example of such relabelling would be a "spatial reversal" transformation, under which all topological relationships involving "in front of" were treated as equivalent to "behind", and so on. This transformation would parallel the "opposites" slippage in Copycat (Hofstadter and Mitchell, 1992) that enabled analogies such as "ABC : ABD is like XYZ : WYZ". An alternative approach to defining transformations is the adjacent node combination of the VAE model of analogical reasoning (Sowa and Majumdar, 2003).

The model does not specify the process by which transformations are constructed, although examples include: 1) transformations are constructed that would improve current highly-performing associations according to the measure $\stackrel{q}{\rightarrow}$, and 2) transformations that have been successful in previous iterations of the model (potentially on different analogy problems with different sources) are re-added to the current set of candidates. The former option requires working backwards from what is necessary to improve current associations, while the latter requires remembering transformations from past problems. The model also does not specify the conditions under which reinterpreting should be initiated – when the current transformation warrants further exploration and when it should be changed. Potential questions include whether seeking a transformation is always appropriate when no associations are found, or whether some objects should be just considered incompatible. While the Idiom model presented in this paper represents one approach to these questions the issue remains an area of active research in interpretation-driven systems.

Transformation evaluation produces an evaluation of all the candidate transformations in T by applying them to the candidate associations in A and measuring their quality with $\stackrel{q}{\rightarrow}$. The specifics of this process are left to an implementation of Idiom, but given the computational complexity of testing for subgraph isomorphisms implementations randomly sampling from A for each transformation seems preferable to comparing all possible pairs of $a \in A$ and $t \in T$. The process for this would involve randomly selecting a subset of "test" candidate associations from A for each candidate transformation in T. This selection could be weighted by the quality measure $\stackrel{q}{\rightarrow}(a), \forall a \in A$, which would already have been calculated during the association evaluation process.

Once an evaluation of the quality of each transformation in T is known one candidate is selected to be τ, the transformation applied to the graph representations to influence mapping. This selection process may be deterministic (selecting the transformation with the highest quality after evaluation), stochastic (selecting a transformation with probability based on quality evaluation), or may involve some other selection heuristic(s).

Transformation application applies the current τ (either new or retained from the previous increment) to the graph representations $g(o_1)$ & $g(o_2)$. After the new transformation is applied this increment of the interpreting process completes and the mapping process takes over to continue the search for mappings.

4 Constructing associations with Idiom

We have developed an implementation of the Idiom model that demonstrates its feasibility and serves as a proof-of-concept of the IDM framework. We present selected results from the implementation showing its capacity for transforming representations and constructing associations.

4.1 Implementing Idiom

The implementation of Idiom described in this paper is in the domain of visual analogy, constructing relationships between images and drawings based on their spatial properties. The Idiom model is more generally applicable than this implementation, but it is intended to serve as a proof-of-concept of both Idiom and the IDM framework from which it is derived. Associations constructed by this implementation will be based on perceptual relationships, such as geometrical or spatial relationships between shapes. This system can perform mapping on shapes similar to those of CogSketch (Lovett et al., 2009b,a; Forbus et al., 2011), but our focus is on representational transformation and the integration of representation and mapping, rather than on relational abstraction. Visual associations are an important component of many design processes and offer a rich variety of potential design domains.

The implementation takes visual representations of objects in the form of 2D vector drawings, extracts features from them, constructs relationships between those features and then makes analogies using those relationships. The definition of a "feature" in this implementation is a minimal closed shape, and features are described using the outline of these shapes, not their weight, colours or other styling. These shape features are categorised into groups based on the similarity of their outlines using a centroid-radii method for shape description proposed by Tan et al. (2003).

The shape-based featural representations are analysed for a variety of topological relationships, including relative scale, orientation, position and the sharing of edges and vertices. Typological relationships include conceptual similarity and conceptual identity, although this implementation uses exclusive concepts – each

feature belongs to exactly one concept. This simplifies the perception process allowing us to focus on the interpreting process that is the focus of the IDM framework. Similar topological relationships are aggregated into discrete categories like "slightly smaller" or "approximately 10° difference in orientation".

The implementation uses a genetic algorithm to search the transformed graph representations for contiguous subgraphs. Based on the approaches used in Wang and Zhou (1997) and Cross et al. (1996), the genotype is a set of node:node connections between object graphs, and the fitness function is based on the number of connections that are valid (our quality measure). Transformations applied to the object graphs change the fitness landscape by affecting the validity of these mappings. An evolutionary computation approach was adopted as it produces populations of candidates simultaneously (for use in interpreting) and can have its objective landscape easily changed between generations. A population of candidate associations between the source and target are created randomly and then evolved, with the quality measure $\stackrel{q}{\rightarrow}$ acting as the fitness. Each candidate association in this population is represented by a set of connections between every feature in the smaller graph to a feature in the larger graph. Many such connections may not be valid mappings, but these are discarded by the quality measure. The quality measure is based on the size of the largest contiguous subgraph validly mapped, i.e. the number of features connected by a pattern of shared relationships in the analogy. While size (in features) is only an approximate measure of analogical quality it is suitable for this proof-of-concept.

Transformations in this implementation use edge label substitution to re-interpret the object graphs. An edge label substitution consists of one or more pairs of edge labels (feature-to-feature relations) that are to be treated as interchangeable for the purposes of the mapping process. This enables interpreting of the kind "treat X in the source object as if it were the same as Y in the target object". This gives more interpretive freedom than the visual analogy approach used in Galatea (Davies and Goel, 2001; Davies et al., 2003; Davies and Goel, 2003), which applies a fixed set of known transformations to elements of an image.

Idiom's search for what transformation to apply is based on syntax, not semantics, using the question "what transformation would improve the mappability of these two objects the most?" This relies on underlying syntactic commonality rather than any semantic content about the relationship between relationships. For example, an object containing a line of shapes of increasing sizes could be associated with another object containing a line of shapes of increasing orientations. The implication of these implementation decisions is that when two patterns of relationship share the same structure, but not the same relationships, a transformation can be constructed to treat them as mappable. The approach is related to the corpus-based label substitution in EMMA (Ramscar and Yarlett, 2003), where similarity along the dimensions that best describe a large dataset of examples is used to substitute contextually similar word. Idiom, by contrast with EMMA, seeks out transformations that can most improve the mappability of the source and target object,

without reference to their similarity.

The Idiom approach has the advantage of greatly relaxing constraints on what transformations are possible when compared to previous systems, which allows the construction of a greater diversity of mappings. It does so at the cost of explicability: transformations are justified based on their effect (enabling a more complete mapping) rather than their meaning (the relationship between the concepts so transformed). See Section 5 for details on how Idiom's re-representation capabilities can be contrasted with other analogy-making systems.

4.2 Example analogies in visual art and ornamental design

A variety of vector representations of visual art objects, ornamental design motifs and architectural objects were drawn from images, taking care to select objects which could be meaningfully represented by lines and shapes. These vector images were provided to the implementation to investigate the feasibility of the Idiom model in a design domain. Several of the resulting associations are presented here, along with descriptions of how they were produced by the model. We demonstrate that without IDM a structure-mapping based system could only have constructed these analogies with specially constructed representations hand-built for the task.

4.2.1 Example 1

Figure 6 shows the first example of the output of the computational implementation of Idiom. The figure depicts the vector representations of Object 1 (o_1), a Hittite sun symbol, and Object 2 (o_2), a French empire motif, both from Humbert (1970). The thick solid lines - both grey and black - indicate the lines that make up the object representation itself, with the black lines indicating object features that are part of the association. The thin solid lines connecting features in Object 1 with features in Object 2 indicate the pairs of features that have been mapped by the successful association a_{win}. The thick black dashed lines joining features within each object indicate the relationships between those features that form the basis for the mapped relationship. Each of these connections is labelled with the relationship it represents. The grey box at the bottom of the figure contains the current value for the active transformation τ.

In Fig. 6, the seven points of the star in o_1 are mapped to the seven petals of one floret in o_1, with adjacent points being mapped to adjacent petals. Each of the adjacent pairs of points in Object 1 are connected by an "approximately 50° difference in orientation" relationship (abbreviated using Δrot in the diagrams), while each of the adjacent pairs of petals in Object 2 are connected by an "approximately 20° difference in orientation" relationship. The association was made in the presence of a transformation, τ, equating these

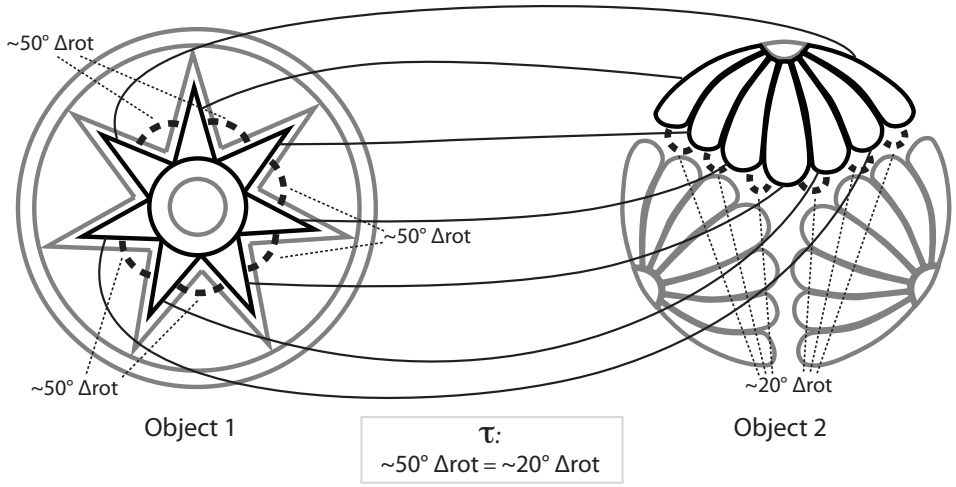

Figure 6: An association produced by Idiom which maps adjacent points of the seven-pointed star in Object 1 to adjacent petals of the top floret in Object 2, based on interpreting the "50° orientation difference" relationships between the star-points to be like the "20° orientation difference" relationships between the petals.

two relationships. Idiom is making the analogy between the star and the floret in the context of treating those two relationships as alike. Idiom possesses no knowledge about the relative similarity or difference of the two orientation relationships, and they could just as easily have been one orientation relationship and one of scale, or any other relational type possessed by Idiom. The transformation that enables this mapping was selected on the syntactic similarity of the representations of the two objects, which each possessed a structurally similar pattern of homogeneously labelled edges. Naive structure-mapping would have failed to construct this association without additional knowledge.

Idiom's process to construct the association seen in Figure 6 began with the system attempting to map between the untransformed representations. $\stackrel{m}{\rightarrow}$ creates (initially by chance) a candidate association, a_1, that maps at least one pair of adjacent star-points to a pair of adjacent petals. a_1 is not of any value by the quality measure $\stackrel{q}{\rightarrow}$ as there is no suitable transformation available to equate the two different orientation labels. $\stackrel{i}{\rightarrow}$ then works backwards to construct a transformation that would increase the value of a_1, and one of the candidate transformations thus constructed is the one shown in Figure 6, which we will call t_1. As features can be mapped using t_1 that cannot be mapped without it, it outperforms other available transformations – including the default null transformation in which the graphs are unchanged – and becomes the active transformation τ. This transforms the graph representations to treat the 20° and 50° orientation relations as alike. $\stackrel{m}{\rightarrow}$ then adopts a search trajectory to maximises the number of associated nodes given $\tau = t_1$. This produces an association like the one in Figure 6.

Theoretically, if the objects were represented in precisely the right way, the association depicted in Figure

6 could be constructed without the use of representational transformations. A relationship like "rotationally adjacent", or "sharing a common axis of rotation" could produce mappable representations of Objects 1 and 2 for which no transformational process was needed. Alternatively, relationships between relationships, such as those used in some Structure Mapping Theory based systems (Falkenhainer et al., 1989; Forbus et al., 1995), could be used to describe the orientation differences in a more general way, and these second-order relationships would be mappable. However these approaches require significant prior knowledge on behalf of the experimenters to construct representations of the two objects that feature compatible relationships. The Idiom method discovers appropriate transformations from regularity in the structure of the object representations, rather than from compatible relational labels.

4.2.2 Example 2

As a stochastic system, Idiom can produce different results when run repeatedly with the same stimuli. Figure 7 shows the result of another run involving the same two objects. In this case six of the seven points of the star in Object 1 are mapped to the two outer-most petals of each floret in Object 2. The difference in orientation between every third star-point (150°) is associated with the difference in orientation across each floret and between each floret, which are both 120°. The analogy-making system has interpreted these two different relationships as being alike, transforming different parts of the object representations to the first example (Figure 6) and producing a different analogy. The parallel interacting processes of $\stackrel{i}{\rightarrow}$ and $\stackrel{m}{\rightarrow}$ uncover and map between intra-object shared patterns of disparate relationships. In the case of Figure 7 this is the pattern of relationships between every third star-point and the pattern of relationships between outermost petals.

For the production of the association in Figure 7 a series of steps occurred that were similar to those described in Figure 6. A partial association was constructed by chance within the model, a transformation was generated from it and then $\stackrel{m}{\rightarrow}$ pursued the best possible association in that interpreted search space. The stochastic processes involved in the search for both candidate associations and candidate transformations led the system to a explore a different trajectory to the first example. This example would also have been impossible to construct via structure-mapping theory without changes in the knowledge structure.

These examples demonstrate that the Idiom model can produce analogies between simple visual objects. The computational implementation of this model has produced multiple varied outputs depicting associations that are different in both appearance and substance. It does this by "forcing" mappings between semantically different relationships based on the underlying syntactic similarities of the objects being mapped, such as the patterns of 120° and 150° relationships transformed in Example 2 and the similar pattern transformed in Example 1. Idiom, and the IDM framework it instantiates, can produce associations that – intuitively, at

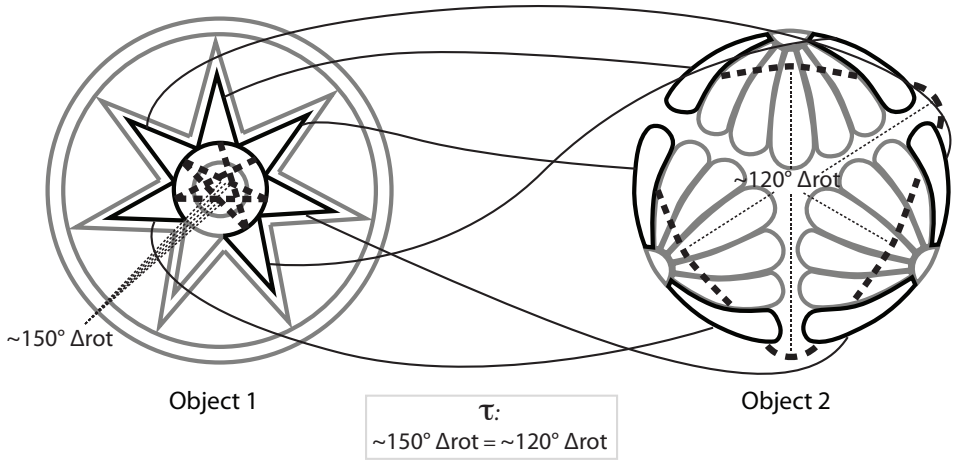

Figure 7: An association produced by Idiom which maps every third point in the seven-pointed star in Object 1 to the set of outermost petals on the three florets in Object 2, based on interpreting the "150° orientation difference" relationships between the star-points to be like the "120° orientation difference" relationships between the petals.

least – seem like those expected by a human observer, such as those between adjacent petals and adjacent star-points in Figure 6. Interpreting can also lead to associations that differ significantly from those likely to be produced by a human (again, intuitively at least), such as the more unexpected association between outermost petals and every third star-point in Figure 7.

4.2.3 Example 3

Among the vector representations of art and design objects provided to the implementation were a line-art version of M. C. Escher's 1938 painting "Sky and Water I" and a nineteenth century wrought ironwork pattern intended for use in a gate (Cottingham (1824) via Cliff (1998)). An association between these two objects can be seen in Figure 8, where the four rows of fish which make up the bottom half of Object 3 have been mapped to the four rows of quadrilateral-like shapes in the middle of Object 4. The features of each row of fish belong to their own conceptual category, as the fish become progressively more abstract towards the centre of the painting, resulting in a "similar concept" relationship existing between adjacent rows. The curved bars which make up the ironwork pattern in Object 4 create a similar pattern, with the gaps between the bars becoming slightly smaller with each successive row. The Idiom implementation uses a transformation that equates these two relationships to map between the fish and the gaps between bars.

Figure 8 shows that Idiom can construct relationships between structurally similar representations even when the relationships involved are disparate and the objects being represented are in different domains. Not only could naive structure mapping not have constructed this association without additional knowledge,

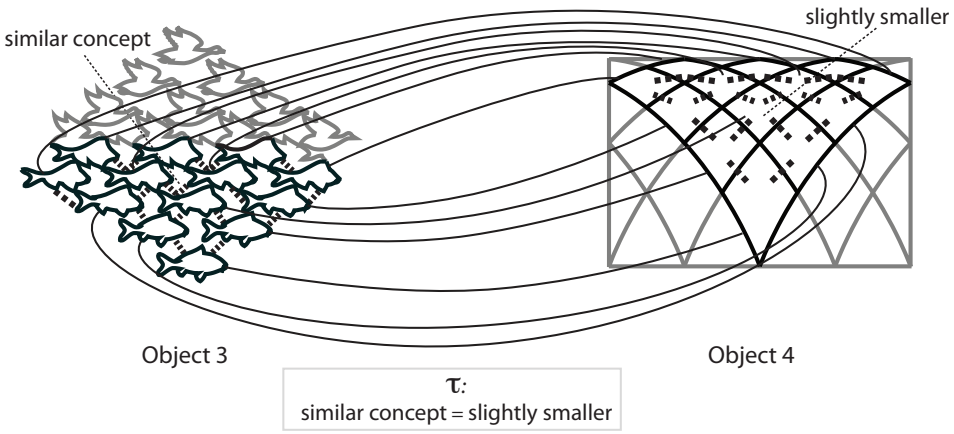

Figure 8: An association produced using Idiom which maps the triangular pattern of increasingly more abstract fish in Object 3 to the triangular pattern of increasingly smaller gaps in the gate in Object 4, based on interpreting the 'similar concept' relationship between the fish to be like the 'slightly smaller' relationship between the gaps.

but it is difficult to conceptualise a representation in which the relationships of similarity and scale are expressed identically. The two relationships are objectively different, but in the context of this association it is of potential benefit to explore what the world would look like if they weren't. Using IDM, our system is capable of mapping a pattern of typological relationships between figures in a painting to a pattern of topological relationships between figures in an ironwork gate. This result demonstrates the possibility of inter-domain association using Idiom given symbolic representations of both objects.

5 An IDM-driven comparison of representation change in models of analogy-making

In addition to serving as a basis for developing models of computational analogy-making, the IDM framework can serve as a lens through which the interpreting capabilities of analogy-making systems can be described. The constituent processes of models of analogy can themselves be analogously mapped to IDM's perception, interpreting and mapping. An analogy-making model can be described using IDM so long as it exhibits a mechanism by which previously unmappable object representations can be transformed so as to make them mappable. By re-framing other systems through this lens we can describe commonalities in the way they transform representations despite the diversity of methods by which they do so. The IDM framework asks six questions of an analogy-making model (see Section 2), and we follow this line of questioning for three extant models and the Idiom model described in this paper. Idiom does not implement transfer and therefore is not a complete model of analogy-making like the three to which it is being compared, but it

does instantiate the mapping and representational change components of the analogy-making process that are under investigation.

The three models used for comparison come from the three paradigms of computational analogy-making models in French (2002): the symbolic, the connectionist and the hybrid. The canonical examples of each (according to French) are the Structure Mapping Engine (SME) (Falkenhainer et al., 1986, 1989), Analogical Mapping by Constraint Satisfaction (ACME) (Holyoak and Thagard, 1989; Spellman and Holyoak, 1992), and Copycat (Hofstadter, 1984; Hofstadter and Mitchell, 1992). However, the SME is a model of analogical *mapping* only and does not incorporate any representation construction or transformation. We compare an extension of it, Incremental Structure Mapping (I-SME) (Forbus et al., 1994) which incorporates a form of reinterpretation into the basic SME model. The incremental nature of I-SME addresses the problem of cognitive plausibility when problem-solvers are exposed to knowledge about potential analogues sequentially, with new knowledge arriving after the mapping process has begun. I-SME adds new knowledge to object representations during mapping, and decides in each case whether to extend the existing mapping or construct a new and different one. We compare I-SME, ACME, Copycat and our Idiom implementation through the lens of the IDM framework.

I-SME is a general theory of analogical mapping, and thus can map between any symbolic representation, but the best-known implementation of it, the Minimal Analogical Reasoning System (MARS) solves engineering problems given a worked example of an analogous problem. This implementation of I-SME combines basic knowledge about objects, incremental mapping and an equation solver to solve problems. It starts with an initial match between the objects in the two problems (the solved source and the unsolved target), and then incrementally extends its representations by generating candidate inferences, evaluating them, and incorporating the successful ones into its mapping. While the creators of I-SME intended it to be able to incorporate new external knowledge (such as that provided by a human teacher) during the course of constructing an analogy, we assume for this comparison that no such external knowledge is available. I-SME's new knowledge will come solely from derivations produced through transfer or deduction.

ACME also maps between symbolic representations of real-world concepts, and is able to construct analogies like "Socrates is a midwife of ideas". ACME operates on graph representations of features and relationships, and constructs mappings by spreading activation through that graph, with simultaneously active concepts being analogous. This activation is guided by a set of constraints governing what kinds of properties should be mapped. ACME adds excitatory and inhibitory connections between features to constrain mappings between those features to be similar in structure, similar in meaning, and relevant to the current context. ACME operates by simulating these constrained networks until they settle into a low energy state, in which simultaneously activated concepts are both structurally consistent and semantically

similar.

Copycat finds analogical connections within the domain of letter strings, solving proportional analogies of the form "$abc \rightarrow abd, ijk \rightarrow$?". Copycat has knowledge of concepts relevant to the domain – such as "group of letters", "identical letters" and "successive letters" – in the form of a connectionist semantic network. Copycat builds representations of letter strings out of these concepts using a parallel representation construction and mapping search process. Copycat operates by a stochastic search of possible representations of the problem affected by both top-down pressures from the conceptual network and bottom-up pressures from the letter strings.

Our implementation of the Idiom model (see Section 4.1) finds analogous patterns of shapes within vector images drawn from ornamental design, architecture and art. Images represented as graphs are iteratively searched for mappings by a genetic algorithm and transformed by an interpreting process driven by reinforcement learning. While our Idiom implementation is the only one of these four systems to have been constructed according to the IDM framework, we demonstrate that IDM framework is sufficiently general as to scrutinise and compare the mechanisms of each system.

The first question posed in the IDM framework is "What bounds the space of valid associations constructible by mapping?", Table 1.

Defining the space of valid associations			
I-SME	**ACME**	**Copycat**	**Idiom**
Features can be associated when they share relations.	Concepts within the graph are associated when simultaneously active.	Strings of letters can be associated when they share representational structure.	Features can be associated between shapes when there is a shared pattern of relationships.

Table 1: Answers to the question: "What bounds the space of valid associations constructible by mapping?".

The mapping search spaces used by I-SME, ACME, Copycat and Idiom all share a commonality: they map only between features that are contextually identical. They differ in their representations, search strategies, and definitions of "context", but each will map patterns of matching features. Copycat, I-SME and Idiom follow the structure-mapping approach of mapping relationships between features rather than features directly. ACME maps features but its process for doing so can be thought of as contextually equating the relationships between them as identicality, and thus can also be thought of as mapping structures of relationships.

The second question posed in the IDM framework is "What causes an association to be selected by mapping?", Table 2.

The four models can be divided into two categories based on how they terminate the map-

Selecting candidate associations			
I-SME	**ACME**	**Copycat**	**Idiom**
Successful associations transfer knowledge that fulfills a goal.	An association is selected when an energy minimum is reached.	An association is selected when a solution to to the proportional problem can be generated.	Associations are selected when the quality measure reaches a pre-defined threshold. In the implementation this is based on the number of mapped nodes.

Table 2: Answers to question: "What causes an association to be selected by mapping?".

ping/interpretation cycle: those that implement transfer that those that do not. The two models that include a transfer component, I-SME and Copycat, terminate when they can successfully transfer knowledge from the source to the target and solve the provided problem. This makes the selection of an association problem-based. The two models that do not include a transfer component, ACME and Idiom, use alternate quality measures. ACME terminates based on local maxima, by reaching a state where no further mapping can occur. Idiom terminates based on a specified suitability criterion that is specified according to the problem domain. The Idiom implementation presented in this paper uses a measure of quality derived from the association structure (size in nodes) as it does not implement a transfer or problem-solving component.

The third question posed in the IDM framework is "What bounds the space of valid transformations constructible by interpreting?", Table 3.

Defining the space of possible transformations			
I-SME	**ACME**	**Copycat**	**Idiom**
Candidate inferences are constructed that influence the next mapping increment.	Constraints apply pressure from contextually related nodes to force disparate nodes to map.	Transformations equate the meaning of disparate concepts and affect the trajectory of representation construction.	Theoretically any graph transformation. Implemented as equivalency between pairs of edge-labels in prototype.

Table 3: Answers to the question: "What bounds the space of valid transformations constructible by interpreting?".

Three of the four systems, ACME, Copycat and Idiom, implement transformation as contextual equivalency between relationships. They differ, however, in how and why these transformations are generated. In ACME and Copycat relational equivalencies are enabled by pre-specified connections between concepts, although in both this affects representations indirectly. In ACME this enabling connection is in the form of constraints favouring certain mappings over others, and in Copycat certain concepts are "slippage"-capable

in specified circumstances. The connectionist nature of ACME means that these equivalencies are specified in a distributed but deterministic fashion, while the hybrid connectionist/symbolic nature of Copycat means that while equivalencies are specified symbolically they only affect representations through parallel distributed (and stochastic) processes. In Idiom the interpreting process is symbolic and stochastic as in Copycat, but all pairwise equivalencies are possible as in ACME.

I-SME differs from the other three systems in that transformation does not take the form of contextual equivalency between features of the object representations. I-SME can receive new knowledge between mapping steps either from an external source like the experimenter or through its own deduction or inference. These transformations do not contextually treat objectively different properties as contextually similar, but instead represent an evolving objective knowledge base. For example, I-SME could be implemented with a set of knowledge about cats (that they are kept as pets by humans, that they have fur, etc) and use that to add knowledge to an object once it had established that the object belonged to the class "cat". Transforming between levels of representational abstraction like this is not the same as contextual equivalency between objects – a cat is not a pet only for the purposes of this mapping, it is always both a pet and a cat.

The fourth question posed in the IDM framework is "What causes a transformation to be selected by the interpreting process?", Table 4.

Selecting candidate transformations

I-SME	**ACME**	**Copycat**	**Idiom**
Successful transformations bring the agent closer to its goal states.	Conceptual activation spreading along constrained relations determines the transformations produced.	Pressures from active representations (bottom-up) and active concepts (top-down) affect what transformations occur.	Selected transformations improve the quality measure $\xrightarrow{q}$. In the prototype implementation this means they enable larger associations.

Table 4: Answers to the question: "What causes a transformation to be selected by the interpreting process?".

Two of the four analogy-making models, I-SME and Idiom, select transformations based on how they improve the associations they act on, while the other two, ACME and Copycat, use contextual influence. I-SME's selected transformations (the inferred attributes that extend its representations) improve candidate associations based on progress towards the goal state, while Idiom's selected transformations improve candidate associations based on the measure of association quality. Both I-SME and Idiom use syntactic properties of the object representations to motivate transformations: the transformations that are generated make candidate associations stronger according to each system's goal. As Idiom can construct any pairwise equivalency between relationships this means it is driven by structural similarity within object relationships – shared patterns of different relationships that can be equated by transformations. I-SME, by contrast, is

limited in its transformations by what it can derive from existing knowledge using rules and inference, and cannot contextually change the meaning of its representations, only add to them.

ACME and Copycat select their transformations using the effect of the current task context. In ACME every node exerts influence – however small – on every other node's activation and thusly what it is analogous to. In Copycat representational elements beget more like them, but there is also influence exerted by active concepts, and active concepts spread their activation to similar concepts in the same way as ACME. Copycat's transformations (referred to as "conceptual slippage") occur only in the right combination of conceptual activations, in contrast to ACME's transformations which occur as the sole, iterative mechanism by which analogies are constructed.

The fifth question posed in the IDM framework is "How do candidate transformations selected by interpreting affect mapping?", Table 5.

Effects of candidate transformations on mapping

I-SME	**ACME**	**Copycat**	**Idiom**
Mapping search applies to transformed objects directly.	Mapping and Interpreting are not separate – associating between two features and transforming them to be equivalent is the same operation.	Transforms the meaning of nodes in its conceptual network, which affect representation construction and thus mapping search.	Mapping search applies to transformed objects directly.

Table 5: Answers to the question: "How do candidate transformations selected by interpreting affect mapping?".

While the cause and scope of their transformations are very different, I-SME and Idiom both use transformations that directly change the representations being searched. This change then affects the search trajectory by re-defining the problem space – what was once mappable may now not be, and vice versa.

ACME and Copycat, while they both use spreading activation among conceptual networks to trigger their transformations, have very different processes for how those transformations affect mapping. In ACME the network of nodes and relationships between them is the only representational structure present, and transforming the meanings of those nodes (expressed as the context of simultaneous activations) is both the mapping process and the interpreting one. This equivalence between transformation and mapping reflects the core of the connectionist approach: "meaning" exists only in context, and thus both transformations of meaning and solutions to problems are embedded in distributed structures. While this may seem to violate a comparison of this system with the others under the IDM framework, note that it is only the last two questions which are difficult to answer in ACME, and their answering still proves valuable for understanding

the system.

The sixth and final question posed in the IDM framework is "How do candidate associations selected by mapping affect interpreting?", Table 6.

Effects of candidate associations on interpreting			
I-SME	**ACME**	**Copycat**	**Idiom**
Candidate associations determine which rules can apply.	Mapping and Interpreting are not separate – associating between two features and transforming them to be equivalent is the same operation.	Candidate associations influence conceptual activation and and thus can bring about transformations.	Candidate associations are used to evaluate transformations, thus affecting which one(s) apply.

Table 6: Answers to the question: "How do candidate associations selected by mapping affect interpreting?".

Each of the four systems provides a very different answer to this question. In I-SME the rules of symbolic logic apply – transformations can only occur when there is a clear logical precedent. In ACME, again, the question does not have a clear meaning, as transformations and association candidates are both expressed as conceptual activation patterns. In Copycat there are indirect effects on transformation brought about by the current candidate associations: existing representational fragments affect conceptual activation, and certain patterns of conceptual activation trigger transformation. In Idiom the relationship is more direct: candidate transformations are evaluated based on how they improve current candidate associations, meaning that the evaluation of the former is contingent on the state of the latter.

6 Comparing representation change in Idiom to models of analogy-making

The comparison in Section 5 illustrates two points. The first is that the IDM framework serves as a useful lens through which to view analogy-making models and compare the ways by which their constituent processes interact. IDM is an effective comparative tool for computational models of analogy-making that have historically been categorised as divergent, particularly along the "symbolic" – "hybrid" – "connectionist" axis. The second point is that the Idiom model occupies a niche distinct from previous computational models of analogy-making. We argue that this niche makes Idiom particularly apt for analogy-making in design.

Idiom would formally be best classified as a symbolic system, although this may be contingent on how the interpreting and mapping processes are implemented as it would be possible to do so in a connectionist way, making Idiom a hybrid system. We use the six questions posed when viewing the four analogy systems

according to the IDM framework to illustrate these differences and their advantages in a model of analogical design reasoning.

Idiom's space of possible associations (Q1) is defined according to Structure Mapping: representational structures connected by relations, with analogical mappings arising from shared patterns of relations between objects. The representations used in Copycat share fundamental similarities with those of Idiom and structure mapping, with features (the letters) mappable when they are connected by relational structures (letter groups, etc). Idiom's selection process for candidate associations (Q2) is likewise comparable to those of other systems – those that feature transfer processes evaluate according to system goals, while those that do not evaluate according to features of mapping. These similarities show that mapping in Idiom and the IDM framework is comparable to the notion of mapping as expressed in other models, and that the same problem (analogy-making) is being solved in each approach.

Where Idiom's approach begins to diverge is in the space of possible representational changes during mapping (Q3) and how those changes are selected (Q4). Symbolic models like I-SME typically utilise logical inference as the mechanism for representational change when transformation during mapping is possible at all. Idiom's approach offers more flexibility than I-SME approach in both the construction and selection of transformations: new transformations can be constructed from syntactical regularities in object representations, and the iterative stochastic transformation evaluation process is less constrained than inference (at least in the absence of new external knowledge). The approaches to transformation used by ACME and Copycat are revealed by the IDM reframing as diametrically opposed: While both model transformations as contextual equivalency, Copycat has few potential transformations ("slippages") while ACME allows any pair of features to transform. Conversely, Copycat's slippages occur only after specific representational and conceptual states have been reached, while ACME's transformations occur constantly as a result of aggregate pressure from the entire representational structure. Idiom, by comparison, permits the same space of possible transformations as ACME (as in theory any two relations can be equated) while offering the same guidance in selecting transformations as Copycat (as transformations are only constructed in the context of specific, favourable representational states). These behaviours have occurred separately in previous models of analogy-making, but in Idiom they occur together. This shows the strength of the iterative and parallel approach to interpreting and mapping: everything can in theory be mapped, but local conditions guide individual decisions about which transformations to apply. This is of utility in models of design reasoning, where the ability to maintain ambiguous representations and only hyper-focus situationally has been identified as a key component of design expertise (Cross, 2004).

The second strength of the IDM framework (and the Idiom realisation of it) is in the centrality of the interactions between mapping and interpreting. Interpreting in Idiom affects mapping (Q5) by transforming

the object representations directly, changing the trajectory of the search as in I-SME and ACME. Mapping in Idiom affects interpreting (Q6) by driving what transformations are constructed – the interpreting process explores how to improve extant candidate associations in the same way that slippage occurs when the conceptual context suggests it would be useful in Copycat. The ability to transform representations (and influence the trajectory of mapping) according to the current state of the mapping process is a noted strength of both Copycat and Idiom. However, Idiom's representational structure is domain-general, and its transformations are syntactically derived, obviating the need to specify a domain- and context-specific set of potential slippages. The IDM approach allows mapping and interpreting to iteratively influence each other in a domain-general model of analogy. Permitting this capability outside of the micro-domains that characterise Copycat (and derived systems) is a significant advancement for analogical design reasoning systems.

7 Conclusion

The way representations change during mapping is a critical component of computational models of analogy. We have developed IDM, a general framework for describing representational transformation within analogy-making. IDM focusses on the way iterative reinterpreting occurs in parallel with the mapping process, and how this process mutually affects and is affected by mapping. This general framework has the capacity to investigate how other models of analogy-making have incorporated these interactions. By representing other analogy-making models in terms of these mechanisms we are able to elucidate the similarities and differences in the way their object representations change during mapping. We argue that this framework is particularly apt for representing analogy problems for computational design, as the situated, constructive and emergent nature of design representations (Gero, 1998) requires an approach that supports representational fluidity.

IDM provides a unifying descriptive architecture for comparing the processes used by diverse computational analogy-making models. Reviews of computational analogy-making spanning the last 25 years (Hall, 1989; French, 2002; Gentner and Forbus, 2011) have identified a number of typicalities in extant models. These include typical processes in models of analogy (including retrieval, re-representation, mapping, transfer, and evaluation), typical components of the representations used in analogy (such as labelled relational structures) and typologies of model architectures (connectionist, symbolic and hybrid). The IDM framework introduces an additional way to describe similarities among analogy-making models: their interpretive capacity and its catalyst(s). It is general enough to usefully describe a broad variety of analogy-making models. Once a system has been thus described, the IDM framework poses questions about the representational processes encapsulated in the model and their interactions with mapping. The answers to these questions provide new insight into how models previously thought of as having incompatible theoretical backgrounds

– such as connectionist ACME and symbolic I-SME – are alike.

We have developed a computational model of the mapping component of analogy-making, Idiom, that instantiates the IDM framework. Idiom is specifically designed around the iterative and parallel interactions between re-representation and the mapping process, and serves as a proof-of-concept of the IDM framework. Idiom, and the prototypical implementation of it in the domain of ornamental design, demonstrate that a focus on the changes in representation that occur during mapping can lead to an expanded capacity for mapping without requiring extensive or specially-built representations. Idiom leverages structural similarities that underly object representations to transform the relational structures within those representations to enable mapping. Two objects that may not share any relations may share a pattern of how those relations occur between their features, and it is from this syntactic level of commonality that Idiom derives its transformations.

We evaluate our model by its generative capability: it can construct associations which previous structure-mapping systems could not construct from the same representations. This capability stems from the parallel interactive interpreting and mapping of the IDM framework. The model is able to construct a variety of associations from the same pair of objects by using different transformations, and able to construct associations between relationships regardless of their similarity, permitting cross-domain analogy. We make no claims as to the quality of the associations produced (beyond the internal evaluation by Idiom of the number of features they can connect), nor of their utility for a particular task: Idiom is a model only of the mapping component of analogy-making and produces associations in the absence of transfer. Idiom and the IDM framework on which it is based represent capabilities that structure-mapping systems did not previously possess. Their extension to a full model of analogy including transfer and evaluation is an area of active research.

To illustrate the capabilities of both the IDM framework and Idiom, we compare our implementation to three analogy-making models, I-SME, ACME and Copycat. Through this comparison, in which all four models were expressed using the IDM framework, similarities and differences among the models can be seen. This demonstrates the descriptive power of the IDM framework and highlights that representational variation during mapping is a central component of analogy-making models. This comparison shows that Idiom exhibits capabilities previously only observed separately in models of analogy-making – specifically the ability to contextually guide the selection of a representational change and the ability for a large space of possible representational changes to be applied. These characteristics, which are made possible by Idiom's ability to learn new transformations based on the structure rather than content of the objects it is mapping, make Idiom particularly suited to the representationally dynamic domain of analogical reasoning in design.

References

Barnden, J. A. and Holyoak, K. J. (1994). *Advances in Connectionist and Neural Computation Theory: Analogy, Methaphor, and Reminding*. Ablex Publishing Corporation.

Bhatta, S. R. and Goel, A. (1997). Learning generic mechanisms for innovative strategies in adaptive design. *The Journal of the Learning Sciences*, 6(4):367–396.

Boden, M. A. (1998). Creativity and artificial intelligence. *Artificial Intelligence*, 103(1):347–356.

Boden, M. A. (2003). *The creative mind: Myths and mechanisms*. Routledge.

Casakin, H. and Goldschmidt, G. (1999). Expertise and the use of visual analogy: Implications for design education. *Design Studies*, 20(2):153–175.

Chalmers, D. J., French, R. M., and Hofstadter, D. R. (1992). High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental & Theoretical Artificial Intelligence*, 4(3):185–211.

Clancey, W. J. (1997). *Situated cognition: On human knowledge and computer representations*. Cambridge University Press.

Cliff, S. (1998). *The English archive of design and decoration*. Thames & Hudson (London).

Cottingham, L. N. (1824). *The Smith and Founder's Director: Containing a Series of Designs and Patterns for Ornamental Iron and Brass Work*. Hullmandel.

Cross, A. D., Wilson, R. C., and Hancock, E. R. (1996). Genetic search for structural matching. In *Computer VisionECCV'96*, pages 514–525. Springer.

Cross, N. (2004). Expertise in design: an overview. *Design studies*, 25(5):427–441.

Davies, J. and Goel, A. K. (2001). Visual analogy in problem solving. In *Proceedings of the 17th international joint conference on Artificial intelligence-Volume 1*, pages 377–382. Morgan Kaufmann Publishers Inc.

Davies, J. and Goel, A. K. (2003). Representation issues in visual analogy. In *Proc. 25th Annual Conf. Cognitive Science Society*. Lawrence Eribaum Associations.

Davies, J., Goel, A. K., and Nersessian, N. J. (2003). Visual re-representation in creative analogies. In *The Third Workshop on Creative Systems. International Joint Conference on Artificial Intelligence*, pages 1–12.

Detterman, D. K. and Sternberg, R. J. (1993). *Transfer on trial: Intelligence, cognition, and instruction.* Ablex Publishing.

Doumas, L. A., Hummel, J. E., and Sandhofer, C. M. (2008). A theory of the discovery and predication of relational concepts. *Psychological Review*, 115(1):1–43.

Dunbar, K. (2001). The analogical paradox: Why analogy is so easy in naturalistic settings yet so difficult in the psychological laboratory. *The Analogical Mind. Perspectives from Cognitive Science, Cambridge Mass*, pages 313–334.

Evans, T. (1964). A heuristic program to solve geometric-analogy problems. In *Proceedings of the 1964 Spring Joint Computer Conference*, pages 327–338. ACM Press.

Falkenhainer, B. (1990). Analogical interpretation in context. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, pages 69–76.

Falkenhainer, B., Forbus, K. D., and Gentner, D. (1986). *The structure-mapping engine.* Department of Computer Science, University of Illinois at Urbana-Champaign.

Falkenhainer, B., Forbus, K. D., and Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41(1):1–63.

Fauconnier, G. and Turner, M. (2003). *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities.* Basic Books.

Fletcher, A. (1964). *Allegory: the Theory of a Symbolic Mode.* Ithaca, NY, Cornell.

Forbus, K., Usher, J., Lovett, A., Lockwood, K., and Wetzel, J. (2011). Cogsketch: Sketch understanding for cognitive science research and for education. *Topics in Cognitive Science*, 3(4):648–666.

Forbus, K. D., Ferguson, R. W., and Gentner, D. (1994). Incremental structure-mapping. In *Proceedings of the sixteenth annual conference of the Cognitive Science Society*, pages 313–318. Lawrence Erlbaum Associates, Inc Hillsdale, NJ.

Forbus, K. D., Gentner, D., and Law, K. (1995). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science*, 19(2):141–205.

French, R. (2002). The computational modeling of analogy-making. *Trends in Cognitive Sciences*, 6(5):200–205.

Gärdenfors, P. (2004). *Conceptual Spaces: The Geometry of Thought.* MIT press.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: a guide to NP-Completeness*. WH Freeman New York.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7:155–170.

Gentner, D. and Forbus, K. D. (2011). Computational models of analogy. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(3):266–276.

Gentner, D. and Holyoak, K. J. (1997). Reasoning and learning by analogy: Introduction. *American Psychologist*, 52(1):32.

Gero, J. S. (1990). Design prototypes: a knowledge representation schema for design. *AI Magazine*, 11(4):26.

Gero, J. S. (1994). Towards a model of exploration in computer-aided design. In *Formal design methods for CAD*, pages 315–336.

Gero, J. S. (1998). Conceptual designing as a sequence of situated acts. In *Artificial Intelligence in Structural Engineering*, pages 165–177. Springer.

Gick, M. L. and Holyoak, K. J. (1980). Analogical problem solving. *Cognitive Psychology*, 12(3):306–355.

Gick, M. L. and Holyoak, K. J. (1983). Schema induction and analogical transfer. *Cognitive Psychology*, 15(1):1–38.

Goel, A. K. (1997). Design, analogy, and creativity. *IEEE Expert*, 12(3):62–70.

Griffith, T. W., Nersessian, N. J., and Goel, A. (2000). Function-follows-form transformations in scientific problem solving. In *22nd Annual Conf. of the Cognitive Science Society*, pages 196–201.

Griffith, T. W., Nersessian, N. J., and Goel, A. K. (1996). The role of generic models in conceptual change. In *Proceedings of the 18th Annual Conference of the Cognitive Science Society*, pages 312–317.

Hahn, U., Chater, N., and Richardson, L. B. (2003). Similarity as transformation. *Cognition*, 87(1):1–32.

Hall, R. P. (1989). Computational approaches to analogical reasoning: A comparative analysis. *Artificial Intelligence*, 39(1):39–120.

Harpaz-Itay, Y., Kaniel, S., and Ben-Amram, E. (2006). Analogy construction versus analogy solution, and their influence on transfer. *Learning and Instruction*, 16(6):583–591.

Hodgetts, C. J., Hahn, U., and Chater, N. (2009). Transformation and alignment in similarity. *Cognition*, 113(1):62–79.

Hofstadter, D. (1984). The copycat project: An experiment in nondeterminism and creative analogies. *MIT Artificial Intelligence Laboratory AI Memo 755*.

Hofstadter, D. R. (2008). *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*. Basic Books.

Hofstadter, D. R. and Mitchell, M. (1992). An overview of the Copycat project. In Holyoak, K. and Barnden, J., editors, *Connectionist Approaches to Analogy, Metaphor, and Case-Based Reasoning*. Ablex.

Holyoak, K. J. (1996). *Mental Leaps: Analogy in Creative Thought*. MIT press.

Holyoak, K. J. (2012). Analogy and relational reasoning. *The Oxford Handbook of Thinking and Reasoning*, pages 234–259.

Holyoak, K. J., Novick, L. R., and Melz, E. R. (1994). *Component Processes in Analogical Transfer: Mapping, Pattern Completion, and Adaptation.* Ablex Publishing.

Holyoak, K. J. and Thagard, P. (1989). Analogical mapping by constraint satisfaction. *Cognitive Science*, 13(3):295–355.

Humbert, C. (1970). *Ornamental Design: Europe, Africa, Asia, the Americas, Oceania: A Source Book With 1000 Illustrations*. Thames & Hudson (London).

Hummel, J. E. and Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review*, 104(3):427.

Kann, V. (1992). *On the approximability of NP-complete optimization problems*. PhD thesis, Royal Institute of Technology Stockholm.

Koestler, A. (1967). *The Act of Creation*. Penguin Books.

Kokinov, B. and Petrov, A. (2001). Integrating memory and reasoning in analogy-making: The AMBR model. *The Analogical Mind. Perspectives from Cognitive Science, Cambridge Mass.*

Lakoff, G. and Johnson, M. (2003). *Metaphors We Live By*. University Of Chicago Press, 2nd edition.

Lovett, A., Gentner, D., Forbus, K., and Sagi, E. (2009a). Using analogical mapping to simulate time-course phenomena in perceptual similarity. *Cognitive Systems Research*, 10(3):216–228.

Lovett, A., Tomai, E., Forbus, K., and Usher, J. (2009b). Solving geometric analogy problems through two-stage analogical mapping. *Cognitive Science*, 33(7):1192–1231.

Mahon, B. Z. and Caramazza, A. (2008). A critical look at the embodied cognition hypothesis and a new proposal for grounding conceptual content. *Journal of Physiology (Paris)*, 102(1):59–70.

McDermott, J. (1979). Learning to use analogies. In *Proceedings of the 6th International Joint Conference on Artificial Intelligence - Volume 1*, pages 568–576, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Penn, D. C., Holyoak, K. J., and Povinelli, D. J. (2008). Darwin's mistake: Explaining the discontinuity between human and nonhuman minds. *Behavioral and Brain Sciences*, 31(2):109–130.

Petkov, G., Vankov, I., and Kokinov, B. (2011). Unifying deduction, induction, and analogy by the AMBR model. In *Proc. 33rd Annu. Conf. Cogn. Sci. Soc.* Erlbaum, Hillsdale.

Qian, L. and Gero, J. S. (1996). Function-behavior-structure paths and their role in analogy-based design. *AIEDAM*, 10(4):289–312.

Ramscar, M. and Yarlett, D. (2003). Semantic grounding in models of analogy: an environmental approach. *Cognitive Science*, 27(1):41–71.

Rissland, E. L. (2006). AI and similarity. *IEEE Intelligent Systems*, 21(3):39–49.

Rittel, H. W. (1987). *The Reasoning of Designers*. IGP.

Robertson, I. (2000). Imitative problem solving: Why transfer of learning often fails to occur. *Instructional Science*, 28(4):263–289.

Schacter, D. L., Norman, K. A., and Koutstaal, W. (2000). The cognitive neuroscience of constructive memory. *False-memory Creation in Children and Adults: Theory, Research, and Implications*, pages 129–168.

Schön, D. A. (1983). *The Reflective Practitioner: How Professionals Think in Action*, volume 5126. Basic Books.

Sowa, J. F. and Majumdar, A. K. (2003). Analogical reasoning. In *Conceptual Structures for Knowledge Creation and Communication*, pages 16–36. Springer.

Spellman, B. A. and Holyoak, K. J. (1992). If Saddam is Hitler then who is George Bush? Analogical mapping between systems of social roles. *Journal of Personality and Social Psychology*, 62(6):913.

Tan, K.-L., Ooi, B. C., and Thiang, L. F. (2003). Retrieving similar shapes effectively and efficiently. *Multimedia Tools and Applications*, 19(2):111–134.

Thagard, P. (1997). *Creative Thought: An Investigation of Conceptual Structures and Processes.* American Psychological Association.

Turney, P. D. (2008). The latent relation mapping engine: Algorithm and experiments. *J. Artif. Intell. Res.(JAIR)*, 33:615–655.

Veale, T. and Hao, Y. (2007). Learning to understand figurative language: From similes to metaphors to irony. In *Proceedings of the 29th Annual Meeting of the Cognitive Science Society.*

Visser, W. (1996). Two functions of analogical reasoning in design: a cognitive-psychology approach. *Design Studies*, 17(4):417–434.

Wang, P. (2009). Analogy in a general-purpose reasoning system. *Cognitive Systems Research*, 10(3):286–296.

Wang, T. and Zhou, J. (1997). Emcss: A new method for maximal common substructure search. *Journal of Chemical Information and Computer Sciences*, 37(5):828–834.

Winston, P. H. (1978). Learning by creating and justifying transfer frames. *Artificial Intelligence*, 10(2):147–172.

Wolstencroft, J. (1989). Restructuring, reminding and repair: What's missing from models of anology. *AI Communications*, 2(2):58–71.

Yan, J., Forbus, K. D., and Gentner, D. (2003). A theory of rerepresentation in analogical matching. In *Proceedings of the 25th Annual Meeting of the Cognitive Science Society.*