AICS-2000 11th Artificial Intelligence and Cognitive Science Conference,
pp 110-119, Aug. 23-25, NUI Galway, Ireland, 2000.

LUDI: A Model for Geometric Analogies using Attribute Matching

*A. Bohan, *D. O'Donoghue,*

Dept. Computer Science, NUI Maynooth, Co. Kildare, Ireland.

**Address for Correspondence: diarmuid.odonoghue@may.ie*

Abstract

We review the work of Evans on graphical proportional analogies, identifying the object mappings that underlie many such comparisons. The limitations of Evans ANALOGY model are investigated. We then establish the role of attributes (colour, shape, pattern etc) in such analogies and identify two distinct mapping algorithms that are required by different classes of geometric analogy problems. We identify the conditions under which the alternate algorithms are required to produce a "best" answer. Finally, we describe a computational model (LUDI) that automatically generates the result for a large number of geometric analogies.

Introduction

The aim of this project is to automatically produce the result of a graphical proportional analogy (geometric analogy). These graphical proportional analogies are possibly more commonly recognised as those that are used in human IQ tests. Geometric analogies are included in IQ tests as they are considered to be problems in which a "*high degree of intelligence for their solution*" is required (Evans, 1967). Any program achieving adequate results on IQ tests, may potentially pass the Turing test. As far as the Turing test is concerned, the program may be indistinguishable from the human. This raises the question of that program having "real" intelligence, a question that has been the subject of huge discussion since Turing introduced it. However, Turing himself dismisses the question *'Can machines think?"* by saying this question is *"too meaningless to deserve discussion*" (in Boden, 1990), and we subscribe to this opinion.

Analogies play a central role in many cognitive processes, and so this is of great relevance to the artificial intelligence community. Our investigation begins with the early work of T. G. Evans and his ANALOGY model (1967). This is a program designed to solve those proportional analogies that are used in intelligence tests. The program will either select one of the (supplied) candidate answers as the correct answer, or all will be rejected identifying a false analogy (in contrast our LUDI model actually generates the answer).

Gentner (1983) introduced the founding research in the area of analogy, and the central role of analogical mapping. She derived a theory, known as the *Structure Mapping Theory* that includes the assertion that analogical mappings are based on predicate structure, rather than objects as had previously been thought. This revolutionary assertion, coupled with her *Systematicity Principle,* has formed the basis of most subsequent work in the area. In part, this project is the modification or update of Evans's work based on systematicity and the subsequent analogy framework. Furthermore, we examine the role of attributes in geometric analogies, basing our solutions strategy upon the mapping and transformation of attribute information.

Proportional Geometric Analogies

An *analogy* is a comparison between two domains, where one domain (the *source*) serves to structure the contents of the second domain (the *target*). Such comparisons are used heavily in learning about new concepts, where the source domain provides a structuring framework for the target. For example, we may view light through the source domain of a wave, thereby highlighting certain properties. Alternatively, we may view light as a particle, thereby highlighting the quantum nature of this form of radiation.

A *proportional analogy* is of the form A:B::C:D, where A, B & C are given and D is derived from applying the transformation derived from A to B, to C. We read such an analogy as A is-to B as C is-to D. Typically, the source domain (A:B) identifies some translation(s), which must then be applied to C, yielding D. Hofstadter and Mitchell (1994) investigated proportional analogies formed from letter strings; such as "*aa:bb :: cc:?*". More complex and ambiguous problems include *"iijjkk:iijjll :: xxyyzz:??"*. A neural network approach to proportional analogies has also been investigated (Jani and Levine, 2000).

Geometric analogies then, are graphical proportional analogies (see figure 1), where each of A, B and C identifies a geometric figure. (All Figures follow the same structure, with 3 boxes containing A, B and C, each containing labelled objects). ANALOGY (Evans, 1969) is a two part algorithm that firstly decomposes graphic images drawn on a unit square into symbolic representations. It then uses these descriptions to identify the required solution from the five alternatives.

Our model accepts symbolic descriptions of each domain - roughly corresponding to part 2 of ANALOGY. Thus, in figure 1, A is represented by the following assertions: `(inside(b,a), circle(a), square(b))` while B is `(above(a,b), circle(a), square(b))`. C then might be represented as `(contained-in(2,1), triangle(1), circle (2))`. We require our model to generate the required result D, from this given information.

There are a number of trivial solutions to such problems that we do not want LUDI to generate. Firstly, the solution D might be an exact copy of B, so every problem could be solved by creating a duplicate of B - regardless of A, C, or any transformations. Secondly, produce no answer based on the logic that A is not identical to C, and thus no transformation can apply. Third, D is an exact copy of C because the A:B transformation might only apply to exact duplicates of A. Such answers would not generally be accepted in an equivalent "human" IQ test, and we do not want LUDI to include such simplistic interpretations of geometric analogy problems. The required solution then is: `(above(1,2), triangle(1), circle(2))`.

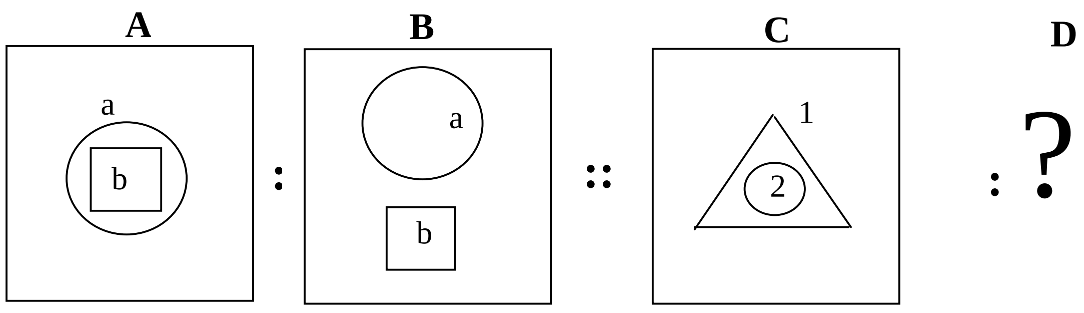

Figure 1: *A simple geometric analogy*

Deriving the correct solution to the problem in Figure 1 (above) necessitates the identification of the following inter-domain mapping: `(inside:contained-in, a:1, b:2)`. A variety of mapping models have been developed to identify the predicate mapping between the source and target domain, and hence the object mapping. These models include SME (Falkenhainer, Forbus and Gentner, 1989), ACME (Holyoak and Thagard, 1990), and IAM (Keane and Brayshaw, 1988). Because there is usually a very high degree of structural similarity between source and target in geometric analogies, the mapping model used by LUDI is of little consequence, but an incremental model was chosen.

Evans ANALOGY program

Because the input to ANALOGY (Evans, 1967) is in the form of line drawings, the program is divided into two parts. Part one decomposes the input figures into subfigures, and various properties and relations between these subfigures are computed. This generates domain descriptions that are composed of dots, straight-line segments and arcs of circles. Relations between the identified objects are also identified, resulting in expressions like `(INSIDE P2 P3)` being generated.

ANALOGY identifies possible matchings between A and B, as it does not know which objects in A relate to which objects in B. One significant difference between ANALOGY and LUDI is that ANALOGY does not recognise the same objects between A and B, treating these as completely different. In contrast, LUDI knows that A and B contain the same objects - the same labels being applied in A and B (see figure 1).

This new information, which is a new description of the input figures, is then passed along to part two. This new information is used to attempt to construct the best 'rule' that transforms Figure A into Figure B, and Figure C into exactly one of the five candidate answers. One of the five possible answers can then be selected using this rule, identifying the correct answer, or alternatively they can all be deemed false. ANALOGY identifies an object mapping between A and B using its *matchab* routine, stating that *"the basis for this matching is the similarity information given as input to part 2"* (Evans, 1967). In contrast, LUDI derives its matching based on the structure of the *predicate* representation (Gentner, 1983).

Attributes in Geometric Analogies

Evans includes few examples of geometric analogies that include attributes, such as that depicted in Figure 3 (Evans's Case 13). Properties of objects, such as the shaded property, are run through part 2 of ANALOGY only, ANALOGY wasn't designed to identify these properties. Little detail is given on how this is achieved - but a single "shaded" attribute is the only one included in his examples. In particular, Evans does *not* explain how to deal with multiple attributes, nor how attribute mappings are identified and used. We shall address this topic in detail in the sections on Local and Global attribute matching.

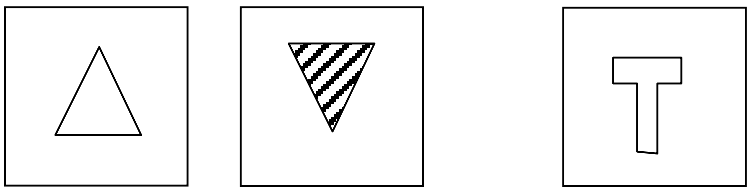

Figure 2: *A Geometric Analogy involving attributes*

LUDI: A Computational Model

We now emphasise the importance given to attributes in proportional analogies. This however contradicts Gentner's Structure Mapping Theory, upon which the foundation of analogy structure is based. Although our theory is essentially based on Gentner's, this treatment of attributes differentiates it from hers. According to Gentner (1983) "*An **analogy** is a comparison in which relational predicates, but few or no object attributes, can be mapped from base to target*". We oppose this dismissal of attributes, and actually place great importance on attribute matching in the derivation of geometric analogies. The attributes of the objects are obviously imperative, in order to distinguish various features such as shape, colour, pattern etc.

We now define two concepts central to the concepts in LUDI. A *mapping* is a source:target pair of concepts that forms the basis of all analogical comparisons (Gentner, 1983). A *transformation* identifies a change in information between A and B, wholly within the source domain. This transformation will later be applied to the target C, generating the solution D. There are two types of transformation; *relational transformation* identifies how relations change between A and B. Figure 3 uses the relational transformation (`inside->below`). An *attribute transformation* identifies an attribute change happening to an object between A and B, in figure 2 the *attribute transformation* (`plain->striped`) is central to generating the required solution.

Firstly, we briefly describe how we identify the inter-domain mapping. This focuses on parts A and C, as these are the only complete sections that can be placed in correspondence. We follow the IAM (Keane et al, 1988) model and identify root predicates, which identifies mapping that are then elaborated. As mapping in such problems is a relatively straight forward task, we shall not dwell on it here. However, we do point out that our mappings are based on predicate structure rather than the object similarity technique used in ANALOGY.

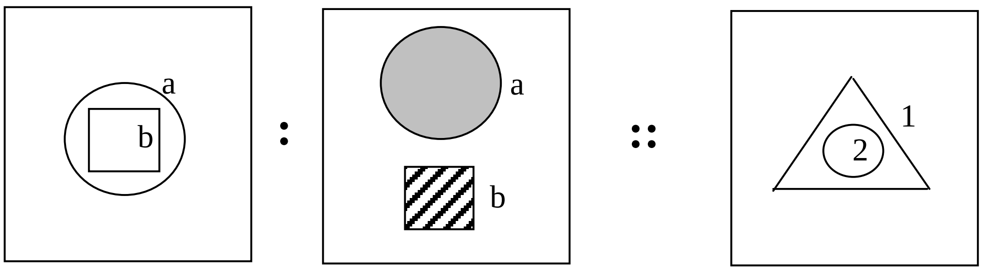

Figure 3: *A geometric analogy requiring with attribute matching*

To support the matching of attributes, LUDI uses a shallow attribute-type hierarchy. The attribute-types supported are; `shape` (square…), `orientation` (face-up…), `colour` (red, white…), and `pattern` (striped, plain…). LUDI allows attribute-matching only between attribute-values of the same attribute type. Thus, in Figure 2 above, the attributes in B (`grey(a)` and `striped(b)`) will be applied to the equivalent objects in the newly generated D. The attribute hierarchy supports objects with multiple attributes (`square, blue, striped, face-down`) etc.

Local Attribute Matching

The majority of geometric analogies can be handled by "local" attribute matching (the only attribute matching example in Evans (1967) is of this type). By this we mean that we identify the required attribute transformation by examining pairs of corresponding objects in isolation, and the attributes of these objects. In figure 4, object `a` is transformed from grey to white, and this is identified without reference to any other objects. This transformation rule is then applied to object `1`, which is the mapped object of `a`. The attribute transformation for object `b` is identified in a similar manner. All examples in this section rely on this local attribute matching scheme. Indeed, in Figure 4 and there is no "general" rule that can be applied to all instances of the `grey` attribute - highlighting the necessity for this local matching scheme.

If there is an attribute transformation between A and B, then applying this transformation to C requires that the mapped object in C has the same attribute as A. In figure 4, object a identifies the rule `grey(a)->white(a)`, which is then applied to the mapped object `1`. A similar transformation rule can be identified for object `b` - without reference to any other objects.

The *null-transformation* condition must also be handled, by identifying that an object has the same attributes in A and B. As such, this null-transformation need not be represented explicitly. Applying this null-transformation to the objects in C leaves them unaltered. Thus, we do not care if null-transform attributes are the same between A and C or not. In figure 4 the shape attribute does not change between A and B, representing a null-transformation condition. Thus, the shape attribute in C is also left unchanged - yielding the required attribute information `circle(1)` and `circle(2)`.

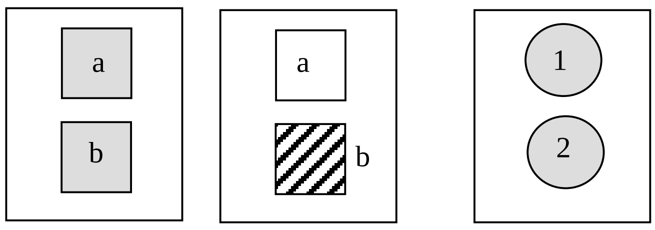

Figure 4: A Simple Local attribute matching problem.

Figure 5 matches multiple attributes (`colour`, `pattern`…) between the source and target. However, the solution can be generated by examining source:target object pairs in isolation and dealing with the attribute transformations one-at-a-time (assuming we have the source:target mapping).

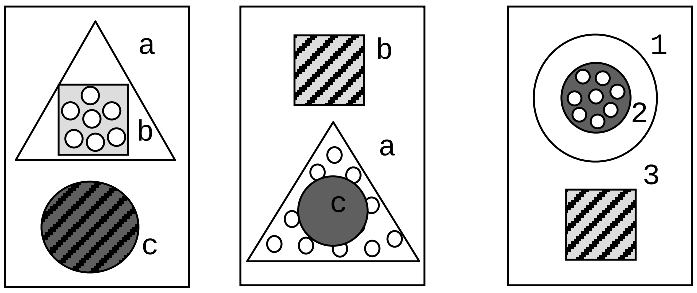

Figure 5: *A Complex Local attribute matching problem.*

Here, the mapping is (`a:1, b:2, c:3`), and we can deal with objects `b` and `2` in isolation. B is light-grey and dotted, becoming light-grey and striped. This transformation can be applied to object `2` with a similar outcome. Obviously, all other predicate and attribute transformations are applied generating the required solution. The important factor is that attribute transformations are identified between isolated object pairs.

Global Attribute Matching

In contrast to local attribute matching, LUDI also solves geometric analogies relying on "global" attribute matching. In figure 6, any attempt to generate a solution based on examining isolated pairs of objects is doomed to failure, and (for this example) it is the `pattern` attribute that generates this ambiguity. This is a direct result of both patterns of `a` (in A and B) and the pattern of `1` in C being different - there is no reasonable basis for either altering, or not altering, the `pattern` of object `1` in the solution.

In fact, LUDI uses this dis-similarity of attribute values between A, B and C to help it determine whether the global or the local attribute matching algorithm should be employed for a particular attribute type. (LUDI's global attribute matching has only been tested on problems where a single attribute type requires the global algorithm, while all other attribute types rely on the local matching algorithm. Because LUDI treats each attribute type independently it could handle multiple global attributes, but this remains as future work).

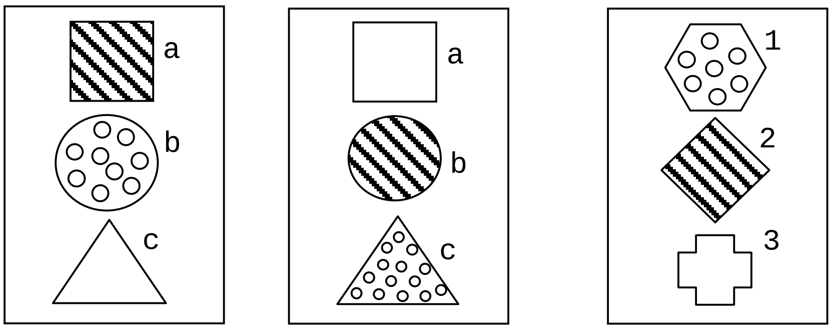

Figure 6: *A Global attribute matching problem*

Rather than isolating object pairs as before, LUDI isolates the "problematic" attribute type (`pattern`) in the source, thereby identifying a global series of attributes and their transformation between A and B. We call this *global* as it deals with all source objects simultaneously (rather than pair-wise). This identifies the global attribute transformation (`striped->plain`, `spotted->striped…`), and when applied to the target generates `striped(1)` and `plain(2)` etc. Ambiguous attribute transformations can occur between A and B, but such problems have ambiguous answers (if any). LUDI does not make an explicit check for such non-analogies. Of course LUDI can also differentiate between the problems (and parts thereof) that require the usual local attribute matching algorithm, and those that require the global attribute matching algorithm.

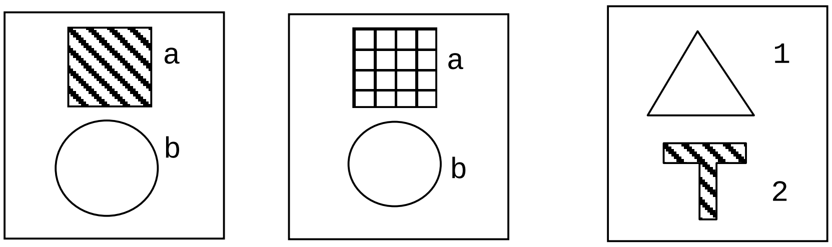

Figure 7: *Another Global Attribute Matching problem*

Consider the result of applying a local attribute matching algorithm to the example in Figure 7. The source domain (A, B) identifies an attribute transformation (`striped->hased`) for object `a`. But the analogy maps `a` to `1`, but object `1` doesn't contain enough attribute information to allow us include the attribute `hashed`. Thus, local attribute matching would generate two plain objects in D. But global attribute matching algorithm allows attribute transformations to be applied that originated from non-mapped objects. The fact that 1 (in A) and 2 (in C) have the same pattern attributes is sufficient to allow the transformation rule to be applied. So global attribute matching correctly generates the target; `above(1,2), triangle(1), t-shaped(2), hased(2)`.

This ability to handle a completely different category of analogy greatly increases the range of problems of that can be handled by LUDI. Furthermore, LUDI identifies that attributes can be dealt with by two very different strategies in geometric analogies, and this perhaps, is partly a justification for Gentner's (partial) dismissal of attributes.

Conclusion

We examine the Evans (1967) ANALOGY model of geometric analogies, but focus on problems that place a much greater emphasis on the attributes of objects. ANALOGY does not address the general problem of identifying an attribute transformation. Furthermore, we require that LUDI actually generates the solution, rather than selecting a solution from five candidate solutions. To support such analogies we utilise a simple attribute-type hierarchy, identifying attribute transformation is the source (A:B) only between attributes of the same type. Such transformations are applied to the target C, generating D.

Significantly, LUDI identifies two different classes of attribute transformation, local and global. The first can be solved by dealing with (source:target) object pairs in turn, while the other deals with attribute types across all source objects. LUDI highlights the importance of attribute matching in geometric analogies.

References

Evans, T.G, *"A Program for the Solution of a Class of Geometric Analogy Intelligence-Test Questions"* in "Semantic Information Processing" (Ed.), M. Minsky, MIT Press, 1967.

Falkenhainer, B. Forbus, Gentner, D. *"The Structure Mapping Engine: Algorithm and Examples"*, Artificial Intelligence, 41, 1-63, 1989.

Gentner, D. *"Structure Mapping: A Theoretical Framework for Analogy"*, Cognitive Science, 7, pg 155-170, 1983.

Hofstadter, D. R. & Mitchell, *M "The Copycat Project: A Model of Mental Fluidity and Analogy-Making"* from Advances in Connectionist and Neural Computation Theory, Volume 2 (Analgoical Connections), edited by Holyoak, K.J. & Barnden, J.A. Ablex Publishing Corporation, 1994.

Holyoak, K.J., & Thagard P. *"Analogical Mapping by Constraint Satisfaction"* Cognitive Science, 13, pg 295-355, 1989.

Jani N, Levine D. *"A neural network theory of proportional analogy-making"*, Neural Networks, Vol. 13, 2, pp. 149-183 (2000).

Keane, M. & Brayshaw, M. *"The Incremental Analogy Machine: A Computational Model of Analogy"* Third European Working Session on Machine Learning. London: Pitman, 1988.

Tomov, N.T. *“Towards Combining Artificially Alive and Artificially Intelligent Agents”* Complexity International, 3, 1996.

Turing, A. *“Computing Machinery and Intelligence”* in The Philosophy of Artificial Intelligence by Boden, M, pg 40-66, Oxford University Press 1990.