Analogy by Abstraction

Stephan Weller (stephan.weller@wiai.uni-bamberg.de)

Ute Schmid (ute.schmid@wiai.uni-bamberg.de)

Cognitive Systems Group
Department of Information Systems and Applied Computer Science
Otto-Friedrich-University
Bamberg

Abstract

We model human solving of proportional analogies of the form $A : B :: C : D$ (where D is to be computed) by the application of E-Generalization. This method allows for the extraction of the common structure of the terms A, B and C and yields a mapping to compute every possible value for D that makes sense with respect to a given background theory. Thus, a formally sound and powerful approach to model human solving of proportional analogies is achieved.

Introduction

Analogy is undoubtedly one of the core features of intelligence. However, few formal descriptions of this process are available. We will consider the special case of proportional analogies, i.e. analogies of the form $A : B :: C : D$ (read: A is to B, as C is to D), where D is to be computed. In this case an algebraic description of the abstraction process can be given by an idea originally developed in [Schmid et al., 2004]. The idea is to apply the method of anti-unification modulo equational theory (E-Generalization) to an algebraic description of the given terms. This method was originally developed in [Burghardt and Heinz, 1996] and [Heinz, 1996] and refined later in [Burghardt, 2005]. By combining these two ingredients, it is possible to extract from the common structure of the two terms A and C a mapping that can be used to generate the solution D from the term B.

The main characteristic of this approach is that direct mapping, as it is used in many cognitive models, is replaced by a mapping via the common abstraction of the base and target. That is, search for a greatest common subgraph of base and target (cf. the systematicity priciple, [Falkenheimer et al., 1989]) is replaced by a mechanism which detects the similarity of objects with respect to the *role* they are playing in the base/target structure. Furthermore, E-Generalization is a elegant way to incoroporate domain-knowledge which can be used to model rerepresentation ([Yan and Gentner, 2003]) in a natural way.

We will first outline a few selected computational approaches to proportional analogy. Then we will introduce the notion of E-Generalization and in a third step show how E-Generalization can be applied to solve proportional analogies.

Computational Approaches to Analogical Reasoning

There exist many approaches to model analogy solving in humans, and some computational models are available. Cognitive models in Psychology include famous examples as SME ([Falkenheimer et al., 1989]), ACME ([Holyoak and Thagard, 1989]), LISA ([Hummel and Holyoak, 1997]), AMBR/DUAL ([Kokinov, 1990] and [Kokinov, 1994]) and IAM ([Keane et al., 1994]). In the following, we will restrict ourselves to some cognitive AI approaches, that are important when dealing with *proportional* analogies. The first two deal with proportional analogies and may be prototypical for other approaches in this domain. The third one is more important to representation issues which will also play a role when solving proportional analogies.

ANALOGY

One of the very early approaches to model analogies is Thomas Evans' program ANALOGY (cf. [Evans, 1968]). It used a microdomain of geometrical figures found in some intelligence tests. All tasks were of the form of a proportional analogy. The answer was to be selected from a list of some possibilities. The selection was based upon an ordering of the five possibilities. The pictures used by ANALOGY were given as Lisp datastructures describing geometrical units like lines, curves and dots. ANALOGY built its *own* representation from the given pictures. However, building up the representation was a separate step and could not be revised while computing the solution.

Although Evans' model was rather primitive compared to more recent approaches, it is still an interesting approach to analogical reasoning, which was unfortunately not really continued by any follow-up projects.

Copycat

One of the most famous approaches to proportional analogies was developed by Hofstadter in 1983 (cf. [Hofstadter and the Fluid Analogies Research Gr., 1995] for an overview). Copycat was developed to find "insightful analogies, and do so in a psychologically realistic way". Hofstadter claims that Copycat is a model of "mental fluidity and analogy making". Copycat works on the string domain, however Hofstadter claims that

the architecture is more general and that Copycats microdomain can model other domains.

The core concept of Copycat is that of "conceptual slippage". This term is used by Hofstadter to note that concepts relatively close to each other may swap their roles when under conceptual pressure.

The main disadvantage of Copycat, and also of other, similar models, is in our view the lack of abstraction in the process of solving an analogy. In humans, it is crucial to see an abstract concept behind the terms to solve an analogy and an adequate model should account for this in some way.

Furthermore, in Copycat a stochastic selection procedure is used at some part of the analogy process. Hofstadter claims that this is an important advantage of his model over others, but in our opinion this is rather a shortcoming, as a stochastic process can not help the understanding of analogy solving in humans.

Structural Information Theory

Structural Information Theory (SIT) was first introduced by [Leeuwenberg, 1971] as a coding system for linear one-dimensional patterns. He represents perceptual structures by three operators named *iteration, alteration,* and *symmetry.* Iteration is supposed to reflect some kind of repetitive process (e.g. $Iter(xy, 3) := xyxyxy$). Symmetry should represent the reversed repetition of a term t after a second term s ($Sym(xyz, ()) := xyzzyx$). Finally, alteration describes the interleaving of a term into a list of terms, such as $Alt(a, (x, y, z)) := axayaz$.

On those operators, Leeuwenberg introduces the notion of *information load*, which is supposed to describe the complexity of an operator. Leeuwenberg claims that the descriptions using the minimal information load correspond to perceptual gestalts (for an introduction to gestalt theory, see [Goldstein, 1980]). His claim is therefore, that the gestalt principle can be explained by even simpler principles, such as his information load.

A more algebraic version of Structural Information Theory can be found in [Dastani et al., 1997]. Here, even some computational modelling of proportional analogies is done.

Syntactic anti-unification and E-Generalization

In this section we will introduce the notion of E-Generalization, as used in [Heinz, 1996] and [Burghardt, 2005]. To facilitate the understanding of E-Generalization, we will first introduce syntactic anti-unification, which is a proper subset of E-Generalization.

Syntactic anti-unification

Unification is a well known and widely used technique in Artificial Intelligence, the probably most prominent application being the programming language *Prolog.* It computes the *most general unifier (MGU)* of two or more terms, i.e. the most general term, such that both terms can be reduced to the *MGU* by applying a substitution. Anti-unification is the dual concept to unification. Instead of computing the most general unifier, it computes the most specific generalization. The result of the anti-unification of expressions $E_1, \ldots, E_n$ is therefore an expression E, such that substitutions $\sigma_1, \ldots, \sigma_n$ exist for which it holds that $E\sigma_i = E_i$ for all $i = 1, \ldots, n$ [1].

In contrast to unification, anti-unification is always possible and there is always a single most specific solution (up to variable renaming).

Algorithms for computing the anti-unification of n terms effectively were introduced by [Plotkin, 1970] and [Reynolds, 1970] independently.

Anti-unification can be used to establish a relation between two terms in the following way: If we anti-unify two terms, and those terms do share some structure, the result will be a non-trivial term containing some variables. Let us for example consider figure 1, an example of anti-unifying two terms that are a formalization of the natural language sentences "John gives Mary a book" and "John gives Peter an apple", respectively. The two terms *gives(john,mary,book)* and *gives(john,peter,apple)* are anti-unified resulting in *gives(john,*x,y*)*. The result of the anti-unification conserves as much of the structure of the terms as possible. The generalized structure reflects the roles the objects are playing in the respective expressions. For example, the variable x describes the role of a receiver that *mary* plays in the first term and *peter* plays in the second one. The second sentence can be obtained from the first one by applying the substitution τ_1 inversely and then applying τ_2.

This is a contrast to a direct mapping approach used in many models for analogies. A direct mapping approach aims at computing the one term directly out of the other, without using intermediate results. This sometimes requires stochastic elements, as used in Copycat ([Hofstadter and the Fluid Analogies Research Gr., 1995]) or the use of heuristics, such as the systematicity principle in SME ([Falkenheimer et al., 1989]). Those may be powerful in solving the analogy, however a stochastical approach is psychologically hardly plausible.

Anti-unification allows for a mapping by using the abstract description of the "roles" some subterms fulfil. In one word, it allows for analogy via abstraction, which has some psychological motivation and also yields a formally sound approach.

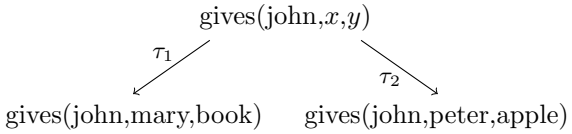

Figure 1: Simple example for syntactic anti-unification

[1] As commonly used $E\sigma$ denoted the application of the substitution σ to the term E

E-Generalization

It is important to note, that while it might be possible to use syntactic anti-unification to model abstraction, there exists one problem. Syntactic anti-unification cannot incorporate any background knowledge. For example, when anti-unifying the terms $5 \cdot 9 + 17$ [2] and $13 + 5 \cdot 7$, one would like a result like $x + 5 \cdot y$. However, as the algorithm cannot "know" that addition an multiplication are commutative, the result obtained will instead be the somehow dissatisfying $x + y$. Here, some kind of background knowledge, in this case some commutative law, should be observed. The incorporation of such background knowledge in the anti-unification process is the aim of E-Generalization. The basic idea is to anti-unify *regular tree grammars* instead of terms.

Regular tree grammars are a language class developed in 1968 (cf. [Brainerd, 1968] and [Thatcher and Wright, 1968]). This language class is located in the Chomsky-Hierarchy between regular and context-free languages (for a very comprehensive introduction to regular tree grammars see [Comon et al., 1997]).

Regular tree grammars allow for the representation of equivalence classes of terms, it would for example be possible to represent the terms $5 \cdot 9 + 17, 9 \cdot 5 + 17, 17 + 5 \cdot 9$, and $17 + 9 \cdot 5$ by one regular tree grammar, assuming commutative laws of addition and multiplication as background knowledge.

The construction of regular tree grammars from a background theory (for example a canonical equational theory) can in some cases be done automatically (cf. [v. Thaden and Weller, 2003] and [Emmelmann, 1991] for criteria when this is possible).

Assuming regular tree grammars for our terms, we can now anti-unify these regular tree grammars by an algorithm originally developed in [Heinz, 1996] and refined in [Burghardt, 2005]. Unfortunately, this algorithm needs exponential time [3] in general, but [Burghardt, 2005] shows that in some cases an efficient computation is nevertheless possible.

It should be noted that the result of this E-Generalization process is not a term, but a regular tree grammar of terms. But this is only a natural consequence of representing equivalence classes of terms as regular tree grammars. The result has to be an equivalence class of terms itself. In the next section we will see, that this will make it possible to compute *all* solutions of a proportional analogy in one step.

For an in-depth description of the algorithm mentioned (and also its implementation and application so proportional analogies) see [Weller, 2005].

E-Generalization may be used as a model for analogies even more than syntactic anti-unification. The features described in the last subsection are fulfilled by E-Generalization as well and additionally, we are not limited to one representation. E-Generalization may account for a change rerepresentation of the terms, which is necessary in many cases (see for example [Yan and Gentner, 2003] for a justification of this claim). We can therefore hope to find a method of solving proportional analogies by E-Generalization. We will describe one method using exactly this idea in the next section, after naming some other applications of E-Generalization.

[2] Standard arithmetic rules are assumed, i.e. $5 \cdot 9 + 17$ is to be read as $+(\cdot(5, 9), 17)$.

[3] exponential in the size of the grammars used

Other Applications

Solving proportional analogies is only one domain where E-Generalization can be applied. There are for example applications in the field of lemma generation (cf. [Heinz, 1996]) or in the completion of number series, as they are used often in intelligence tests. The latter were also worked on by [Hofstadter and the Fluid Analogies Research Gr., 1995] (cf. chapter 2.2), also making use of analogies. The application of E-Generalization to this problem has been done in [v. Thaden and Weller, 2003].

Solving Proportional Analogies by E-Generalization

In the following subsection we will show how to apply the method of E-Generalization to solve a proportional analogy of the form $A : B :: C : D$ (read: A is to B as C to D), where D is to be computed. For the sake of simplicity we will use solely the string domain for our examples, however, anything that can be represented as strings (for example geometrical figures, cf. [Leeuwenberg, 1971] and [Dastani et al., 1997]) could be used. Thus, this is a rather weak restriction [4].

As an example, let us assume we want to solve the proportional string analogy $abc : abd :: ihg : D$. where D is unknown. Using a representation language similar to *SIT* ([Dastani et al., 1997]) we could represent the term abc as $Iter(a, succ, 3)$, meaning that abc is established by iterating the successor operation three times on the constant a. Another possible representation would be of the form $a \cdot succ(a) \cdot succ(succ(a))$, where $\cdot$ means concatenation and *succ* is the successor relation.

Our first aim is now to compute the common structure of the terms A and C, or, in our example the terms abc and ghi. At this point it should be noted, that this common structure could be extracted straightforward by syntactic anti-unification, *if we had knowledge about the structure of the terms* abc *and* ghi. Let us for a moment assume, that we know that the structure of our both terms is $Iter(a, succ, 3)$ and $Iter(i, succ, 3)$ respectively [5]. In this case, an application of syntactic anti-unification would yield the common structure $Iter(x, y, 3)$ and the two substitutions

[4] Stronger restrictions result from the fact that the background knowledge has to be represented as a canonical equational theory, which is not always possible (for criteria cf. [Heinz, 1996])

[5] Actually, the example is not completely formally correct, as it intermixes first and second order terms for *succ* and *pred*, which is of course not valid, but simplifies the example a lot.

$\tau_1 = \{x \leftarrow a, y \leftarrow succ\}, \tau_2 = \{x \leftarrow i, y \leftarrow pred\}$. Let us further assume the structure $Iter(a, succ, 2) \cdot succ(succ(succ(a))$ for the B-term abd. Given this, we could apply τ_1 inversely to the B-term, yielding a new term Q of the form $Iter(x, y, 2) \cdot y(y(y(x)))$. Applying τ_2 to this term would yield the result $Iter(i, pred, 2) \cdot pred(pred(pred(i)))$ which describes the term ihf, which is one possible solution of the analogy.

Seeing this, one possibility to solve a proportional string analogy would be to compute some representation of the participating terms and using syntactic anti-unification (and inverse and normal substitution application) to compute a result. The decision on some representation will thus determine, which result we will obtain.

But instead of choosing one particular representation at the start, we can take the process one step further and use E-Generalization instead of syntactic anti-unification. The complete process is shown in figure 2.

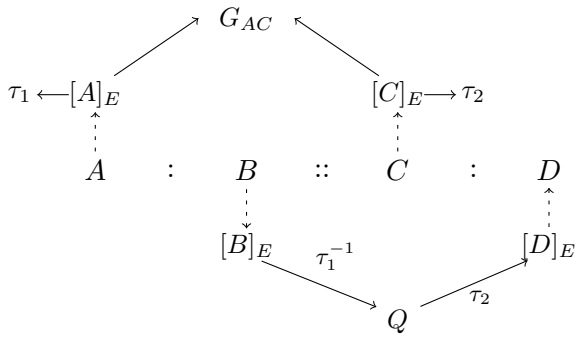

Figure 2: Solving a proportional string analogy

To every ground term A, B, and C a regular tree grammar representing the equivalence class of all representations of the term is built up. Those regular tree grammars are denoted by $[A]_E, [B]_E$, and $[C]_E$. The process of building the regular tree grammars is denoted by the dotted arrows. Next, the E-Generalization algorithm from [Burghardt, 2005] is used on the two regular tree grammars $[A]_E$ and $[C]_E$, thus extracing their common structure as a regular tree grammar G_{AC}. This grammar is not needed in the further process, it is a byproduct of the algorithm, however, it forms indeed a form of abstraction from the ground terms. What *is* needed in the next step, are the substitutions τ_1 and τ_2 also produced by the E-Generalization step.

The substitution τ_1 is inversely applied to $[B]_E$ [6] , resulting in a regular tree grammar describing the common structure between B and D. It is denoted by Q in the figure. Again, this grammar forms an abstraction from the actual terms that can be seen as a byproduct of the analogical process.

[6]Note that (inverse) application of substitutions is well-defined on regular tree grammars. It is a special case of an inverse tree homomorphism. See [Comon et al., 1997] for details.

Finally, the substitution τ_2 is applied to Q, leading to a final grammar $[D]_E$ with the following properties:

- It shares the structure of B, as it is derived by application of (inverse) substitutions to $[B]_E$.
- The constants occuring in the term A are replaced by those in C, as the two (inverse) substitutions τ_1^{-1} and τ_2 are applied.
- It can thus be described as the result of "doing the same thing" to B as it was done to A to get C.
- And therefore, it can be seen as a valid solution of the proportional analogy $A : B :: C : D$.

As mentioned before, the result of this process is not a single term, but a regular tree grammar and as such a whole set of terms. However, it describes *every* result that can be obtained by replacing constants in any representation of B by their counterparts with respect to every possible representation of A and C.

Naturally the question arises, how to extract a term from this tree grammar (when a single term is desired rather than a set of terms). This process is an enumeration of the regular tree grammar, which is in general not possible completely, as infinitely many terms are described by the grammar (imagine for example a grammar for arithmetical operations allowing for the addition of 0 - this alone leads to infinitely big result grammars in nearly every case). Nevertheless, it may be desirable to enumerate the first n terms according to some ordering relation. The enumeration is not a problem, if the ordering relation is defined. But finding a suitable ordering relation is not a simple task. However, using simple relations, like ordering by number of occuring constants, "depth" of the term etc. are possible. Those might even represent some kind of simplicity used by humans to decide which answer to choose in solving a proportional analogy.

Which ordering relation would correspond to the preference humans use is an empirical question and probably requires deep insight in the cognitive processes used in analogical thinking.

Implementation

As mentioned above, the E-Generalization algorithm is exponential in the size of the grammars used in the general case. Thus, no efficient implementation can be expected. Nevertheless, a proof-of-concept implementation of the algorithm was done, which can of course only be used for very "small" examples. For an application to a task which requires more than very small grammars, some restriction to the algorithm is inevitable. Such restrictions would of course depend on the application.

A more extensive description of the implementation can be found in [Weller, 2005].

Conclusion and further work

We have introduced the idea of anti-unification and its extension with background knowledge to E-Generalization. Then we have shown how this method

can be applied to solve proportional string analogies in a generic way, that is, without incorportating domain knowledge into the algorithm.

An important property of this method is the calculation of the common structure of the terms as a byproduct of the analogy solving. In contrast to most other models of analogies, this method does account for the emergence of abstract knowledge without any extra computation. The creation of the abstract knowledge about the common structure is not gained by an extra step, but rather as an intrinsic property of the process.

At least in this aspect this is similar to the way humans solve analogies. One cannot "suppress" the abstraction from the concrete terms. To learn something about the common structure of the terms is inherent in the process of solving the analogy.

To investigate further in human solving of proportional analogies, empirical research is necessary. Our next step will therefore be to conduct an empirical study. The aim of this study will be to check wether the results chosen by humans correspond to a certain ordering relation of the terms in the computed grammar or wether terms occur not covered by the grammar at all (this can of course not be ruled out, as human decisions might not be explicable by background knowledge but rather based on intuition or other non-rational processes).

Acknowledgments

We would like to thank Jochen Burghardt for his support of our work.

References

[Brainerd, 1968] Brainerd, W. (1968). The minimalization of tree automata. *Information and Control*, 13:484–491.

[Burghardt, 2005] Burghardt, J. (2005). *E*-generalization using grammars. *Artificial Intelligence Journal*, 165(1):1–35.

[Burghardt and Heinz, 1996] Burghardt, J. and Heinz, B. (1996). Implementing Anti-Unification Modulo Equational Theory. Technical report, GMD - Forschungszentrum Informationstechnick GmbH.

[Comon et al., 1997] Comon, H., Dauchet, M., Gilleron, R., Jacquemard, F., Lugiez, D., Tison, S., and Tommasi, M. (1997). Tree automata techniques and applications. Available on: `http://www.grappa.univ-lille3.fr/tata`. release October, 1rst 2002.

[Dastani et al., 1997] Dastani, M., Indurkhya, B., and Scha, R. (1997). An Algebraic Approach to Modeling Analogical Projection in Pattern Perception. In *Proceedings of Mind II*.

[Emmelmann, 1991] Emmelmann, H. (1991). Code Selection by Regularly Controlled Term Rewriting. In *Proc. of Int. Workshop on Code Generation*.

[Evans, 1968] Evans, T. G. (1968). A Program for the Solution of a Class of Gemetric-Analogy Intelligence-Test Questions. In Minsky, M., editor, *Semantic Information Processing*, chapter 5, pages 271–353. MIT Press.

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

[Goldstein, 1980] Goldstein, E. B. (1980). *Sensation and Perception*. Wadsworth Publishing Co., Belmont, California.

[Heinz, 1996] Heinz, B. (1996). Anti-Unifikation modulo Gleichungstheorie und deren Anwendung zur Lemmagenerierung. Technical report, GMD - Forschungszentrum Informationstechnick GmbH.

[Hofstadter and the Fluid Analogies Research Gr., 1995] Hofstadter, D. and the Fluid Analogies Research Gr. (1995). *Fluid Concepts and Creative Analogies*. BasicBooks.

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

[Hummel and Holyoak, 1997] Hummel, J. and Holyoak, K. (1997). Distributed representations of structure: a theory of analogical access and mapping. *Psychology Review*, 104:427–466.

[Keane et al., 1994] Keane, M., Ledgeway, T., and Duff, S. (1994). Constraints on analogical mapping: A comparison of three models. *Cognitive Science*, 18:387–438.

[Kokinov, 1990] Kokinov, B. (1990). Associative memory-based reasoning: some experimental results. In *Proceedings of the 12th Annual Conference of the Cognitive Science Society*. Lawrence Erlbaum Associates.

[Kokinov, 1994] Kokinov, B. (1994). The context-sensitive cognitive architecture DUAL. In *Proceedings of the 16th Annual Conference of the Cognitive Science Society*, pages 502–507. Lawrence Erlbaum.

[Leeuwenberg, 1971] Leeuwenberg, E. (1971). A perceptual coding language for visual and auditory patterns. *American Journal of Psychology*, 84:307–349.

[Plotkin, 1970] Plotkin, G. (1970). A note on inductive generalization. In *Machine Intelligence*, volume 5, pages 153–163. Edinburgh University Press.

[Reynolds, 1970] Reynolds, J. (1970). Transformational Systems and the Algebraic Structure of Atomic Formulas. In *Machine Intelligence*, volume 5. Edinburgh University Press.

[Schmid et al., 2004] Schmid, U., Gust, H., Kühnberger, K.-U., and Burghardt, J. (2004). An Algebraic Framework for Solving Proportional and Predictive Analogies. In Schmalhofer, F., Young, R., and Katz, G., editors, *Proceedings of the European Conference on Cognitive Science, Osnabrück*.

[Thatcher and Wright, 1968] Thatcher, J. and Wright, J. (1968). Generalized finite automata theory with an application to a decision problem of second–order logic. *Mathematical Systems Theory*, 2(1).

[v. Thaden and Weller, 2003] v. Thaden, M. and Weller, S. (2003). Lösen von Intelligenztestaufgaben mit E-Generalisierung (Solving intelligence tasks by E-Generalization). In *Tagungsband der Informatiktage 2003*, pages 84–87. Gesellschaft für Informatik e.V.

[Weller, 2005] Weller, S. (2005). Solving Proportional Analogies by Application of Anti-Unification modulo Equational Theory. Bachelor's Thesis, unpublished.

[Yan and Gentner, 2003] Yan, J. and Gentner, D. (2003). A theory of rerepresentation in analogical matching. In *Proc. of the 25th Annual Conference of the Cognitive Science Society*, Mahwah, NJ. Erlbaum.