Syntactic Principles of Heuristic-Driven Theory Projection

Angela Schwering, Ulf Krumnack, Kai-Uwe Kühnberger, Helmar Gust

University of Osnabrück, Germany

Abstract

Analogy making is a central construct in human cognition and plays an important role to explain cognitive abilities. While various psychologically or neurally inspired theories for analogical reasoning have been proposed, there is a lack of a logical foundation for analogical reasoning in artificial intelligence and cognitive science. We aim to close this gap and propose Heuristic-Driven Theory Projection (HDTP), a mathematically sound framework for analogy making. HDTP represents knowledge about the source and the target domain as first-order logic theories and compares them for structural commonalities using anti-unification. The paper provides an overview of the syntactic principles of HDTP, explains all phases of analogy making on a formal level, and illustrates these phases with examples.

Key words: analogy, logic based analogical reasoning, analogical learning, anti-unification

1 Introduction

Analogy making can be considered as a core of cognition: it seems to be central to many cognitive abilities such as organization and retrieval of knowledge, learning new knowledge via abstraction or via analogous comparisons, creativity in finding novel solutions to problems etc. While various psychologically or neurally inspired theories for analogical reasoning have been proposed, there is a lack of a logical theory for analogical reasoning in artificial intelligence and cognitive science. We aim to close this gap and propose Heuristic-Driven Theory Projection (HDTP), a mathematically sound framework for analogy making. In the tradition of classical artificial intelligence, HDTP is based on

a first-order logical language. A logical specification has several advantages for symbolic analogy models, since it can be combined with classical reasoning mechanisms. Drawing logical inferences allows for a new perspective of automatically re-representing knowledge in a way that analogous structures become obvious. While classical reasoning mechanisms have been criticized for being too restricted [28], HDTP extends these reasoning mechanisms by cognitively plausible analogical inferences.

This paper extends a series of papers [17,31,40] and describes the syntactic principles of HDTP. The remainder of the paper is structured as follows: in section 2 we give a short overview of analogies and the various phases of the analogy making process. Section 3 defines the logical language and explains how domain knowledge is represented as a logical theory in HDTP. The analogical mapping between source and target domain is established via anti-unification, which is described in section 4. Section 5 shows how domain knowledge can be re-represented automatically and section 6 explains the analogical transfer in HDTP. In section 7, we discuss analogies with respect to cognitive abilities and show how HDTP can be used to explain certain cognitive phenomena. Section 8 relates HDTP to other (mainly symbolic) analogy models. Section 9 concludes the paper and gives directions for future work. In appendix A, we formalize additional analogies within HDTP.

2 Analogies and Analogy Models

Analogy making is a highly sophisticated cognitive process in which two conceptualizations – specifying a source and a target – are analyzed for common structural patterns [11]. In analogies, source and target are typically of different domains. The purpose of analogy making is to adapt knowledge available from the source domain, such that it can be applied to the target in a way that a new analogous relation can be established between source and target. Analogy making requires intelligence, since analogous patterns and transfers often are not obvious and depend on a certain conceptualization of the domains. Establishing an analogical relation between two domains is furthermore not a deterministic process: there might exist different analogies, which might all be plausible (to a certain extent). Last but not least, analogies often cover the domains only partially or under a certain perspective.

The process of analogy making can be subdivided into several, interrelated tasks. There are distinctions including different pre- and post-mapping phases [6,21,30], however the following three phases are typically considered as core of analogy making: *retrieval*, *mapping*, and *transfer*.

At the beginning, when exposed to a new situation (the target), a source

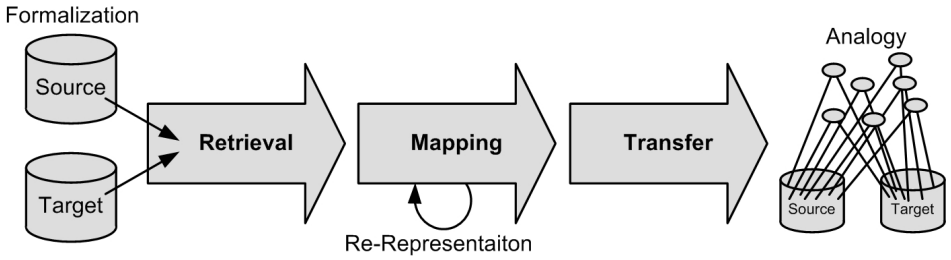


Fig. 1. Phases of analogy making

domain is identified to which that situation can be related. Some retrieval technique has to be applied to search the memory for items which seem like candidates for an analogy. In certain settings, such as intelligence tests or teaching situations, the source domain may be given explicitly. Some models for analogy making view this as the standard case and do not provide special means of retrieval. At the current state of development, this is also the case in the analogy model HDTP presented in this paper. Future research will investigate the application of superficial similarity measures for retrieval similar to ideas implemented in MAC/FAC [12]. In HDTP, superficial similarity can be determined via the number of common symbols in the source and target domain formalizations.

The mapping phase aims to establish an analogical relation between source and target, i.e. it aims to align structures from both domains. In general, there are many possible mappings and which one is appropriate depends on the context and the goal of the analogy. Two problems are typically associated with the mapping step: The *relevance problem* addresses the identification of parts of the domains that are relevant within the context of the analogy and therefore shall enter the analogical relation. The *representation problem* is concerned with the difficulties in mapping, mainly caused by differently structured representations of the domains. While it seems plausible that two domains are represented in an isomorphic way when they are specifically prepared for the analogy, this seems unlikely in general. Therefore the mapping may guide a restructuring of one or both domains. A good deal of the explanatory and creative power of analogies can actually be ascribed to that process of re-representation (cf. section 5).

During the transfer phase the analogical relation is used to translate information between the two domains. Typically, knowledge is transferred from the source to the target domain and is used there to introduce new concepts or structures, provide new explanations to phenomena, or solve problems. This new knowledge is in no way logically justified and should merely be seen as a hypothesis, but it can be the source of valuable inspiration.

In some cases, these three phases are supplemented by additional steps, such as evaluation of the transferred knowledge, or the induction of generalized

knowledge, depending on the model applied and the context in which analogy making is placed. In this paper, we focus on the core phases of analogy making only.

3 Modeling Domains in a Logical Framework

HDTP is a symbolic analogy model based on first-order logic and reasoning techniques. The basis of all operations and processes in HDTP is the formalization of the source and the target domain as sets of many-sorted first-order formulas. First, we provide the formal definition of the underlying language.

Definition 1 (Signature) *A* many-sorted signature $\Sigma = \langle Sort_\Sigma, Func_\Sigma, Pred_\Sigma \rangle$ *is given by a partially ordered set of sorts* $Sort_\Sigma$*, a set* $Func_\Sigma$ *of function symbols* $f : s_1 \times \ldots \times s_n \rightarrow s$ *and a set* $Pred_\Sigma$ *of predicate symbols* $p : s_1 \times \ldots \times s_n$*.*

A signature provides the vocabulary that is used to describe a domain and comprises the following elements: *constants* (i.e. function symbols of arity 0) describe individuals in the domain. Each constant is of a certain sort. *Sorts* [1] describe the type of an entity at a general level and will be used in the analogical mapping process to restrict preferred alignments. Sorts can be interpreted as high-level concepts of an ontology, e.g. *object*, *massterm*, or *integer*. Function symbols are used to represent *functions* mapping individuals to individuals. Predicates express *relations* between individuals. The following two definitions specify the formation of complex expressions:

Definition 2 (Terms) *Assume a signature* Σ *is given. We define the* term algebra $Term(\Sigma, \mathcal{V})$ *relative to an (infinite) set of sorted variables* $\mathcal{V} = \{x_1 : s_1, x_2 : s_2, ...\}$ *with* $s_i \in Sort_\Sigma$ *and a function* $sort_\Sigma : Term(\Sigma, \mathcal{V}) \rightarrow Sort_\Sigma$*. The set* $Term(\Sigma, \mathcal{V})$ *is defined as the smallest set such that the following conditions hold:*

(1) If $x : s \in \mathcal{V}$*, then* $x \in Term(\Sigma, \mathcal{V})$ *and* $sort_\Sigma(x) = s$*.*
(2) If $f : s_1 \times \ldots \times s_n \rightarrow s \in Func_\Sigma$*, and* $t_i, \ldots t_n \in Term(\Sigma, \mathcal{V})$ *with* $sort_\Sigma(t_i) = s_i$ *for* $i = 1, \ldots, n$*, then* $f(t_1, \ldots, t_n) \in Term(\Sigma, \mathcal{V})$ *and* $sort_\Sigma(f(t_1, \ldots, t_n)) = s$*.*

Definition 3 (Formulas) *Let* Σ *be a signature. Define the set of* formulas $Form(\Sigma, \mathcal{V})$ *over* Σ *relative to an (infinite) set of sorted variables* $\mathcal{V}$ *as the smallest set such that the following conditions hold:*

[1] In order to simplify the readability of formal expressions we will suppress the coding of the corresponding sorts, if sort restrictions are clear from the context.

(1) *If* $p : s_1 \times s_2 \times \ldots \times s_n \in Pred_\Sigma$, *and* $t_1, \ldots, t_n \in Term(\Sigma, \mathcal{V})$ *with* $sort_\Sigma(t_i) = s_i$ *for* $i = 1, \ldots, n$, *then* $p(t_1, \ldots, t_n) \in Form(\Sigma, \mathcal{V})$.
(2) *If* $\alpha, \beta \in Form(\Sigma, \mathcal{V})$, *then* $\neg\alpha$, $\alpha \vee \beta$, $\alpha \wedge \beta$, $\alpha \rightarrow \beta$, $\alpha \leftrightarrow \beta$, $\exists x_i : s_i\, \alpha$, $\forall x_i : s_i\, \alpha \in Form(\Sigma, \mathcal{V})$.

Logical operators (such as $\wedge$ and $\neg$) and the *quantifiers* $\forall$ and $\exists$ are available to construct complex facts and rules. A domain can be described by a finite set of formulas. We will call such a set an axiomatization, if it is logically consistent. All formulas that can be inferred from the axioms constitute the domain theory. There may be different equivalent axiomatizations for a given domain.

Figure 2 shows such an axiomatization of a source and a target domain in HDTP [2]: As a running example we use the Rutherford analogy between the solar system and an atom. The source domain represents knowledge about the solar system, stating that the mass of the sun is greater than the mass of a planet (α_1), that there is gravitation between the sun and the planet (α_3), and that for every pair of objects with gravitation between them, the lighter one will revolve around the heavier one provided a positive distance between the objects is preserved (α_6). Rutherford put these facts in alignment with his knowledge about an atom: on the target side, it is known that the lightweight electrons are attracted by the nucleus due to Coulomb force (β_3) and that, despite this attraction, atoms do not collapse (β_4). The latter fact, namely that electrons and nucleus have a distance greater than zero is an abstract formulation of the result of the gold foil experiment due to Rutherford [39].

Due to its first-order representation of domain knowledge, HDTP has a greater expressive power compared to other analogy models: not only a certain situation can be represented, but also general laws like *every two bodies with positive mass will attract each other*. Moreover, a logic-based representation allows for reasoning on formulas and for a re-representation of knowledge.

4 Analogical Mapping via Anti-Unification

If a relevant source domain has been identified for a particular target problem, the analogy is established by comparing source and target for structural commonalities. The analogical mapping phase aims at aligning analogous terms and formulas, i.e. it aims at establishing an analogical relation between source and target domain.

[2] This axiomatization in classical first-order language is translated into a PROLOG notation to serve as input for our implementation.

Solar System	Rutherford Atom
sorts $real$, $object$, $time$ **entities** $sun : object$, $planet : object$ **functions** $mass : object \to real \times \{kg\}$ $dist : object \times object \times time \to real \times \{m\}$ $force : object \times object \times time \to real \times \{N\}$ $gravity : object \times object \times time \to real \times \{N\}$ $centrifugal : object \times object \times time \to real \times \{N\}$ **predicates** $revolves\_around : object \times object$ **facts** $\alpha_1 : mass(sun) > mass(planet)$ $\alpha_2 : mass(planet) > 0$ $\alpha_3 : \forall t : time : gravity(planet, sun, t) > 0$ $\alpha_4 : \forall t : time : dist(planet, sun, t) > 0$ **laws** $\alpha_5 : \forall t : time,\ o_1 : object,\ o_2 : object :$ $dist(o_1, o_2, t) > 0 \wedge gravity(o_1, o_2, t) > 0$ $\to centrifugal(o_1, o_2, t) = -gravity(o_1, o_2, t)$ $\alpha_6 : \forall t : time,\ o_1 : object,\ o_2 : object :$ $0 < mass(o_1) < mass(o_2) \wedge dist(o_1, o_2, t) > 0 \wedge$ $centrifugal(o_1, o_2, t) < 0$ $\to revolves\_around(o_1, o_2)$	**sorts** $real$, $object$, $time$ **entities** $nucleus : object$, $electron : object$ **functions** $mass : object \to real \times \{kg\}$ $dist : object \times object \times time \to real \times \{m\}$ $coulomb : object \times object \times time \to real \times \{N\}$ **facts** $\beta_1 : mass(nucleus) > mass(electron)$ $\beta_2 : mass(electron) > 0$ $\beta_3 : \forall t : time : coulomb(electron, nucleus, t) > 0$ $\beta_4 : \forall t : time : dist(electron, nucleus, t) > 0$

Fig. 2. HDTP formalization of the solar system and the Rutherford atom (Domain-independent predicates such as $>$: $real \times real$ are not listed).

A crucial idea of HDTP is the establishment of an analogical relation via a generalization of the source and target domain. HDTP applies the formalism of anti-unification to compare formulas of the source and the target theory for structural commonalities. Anti-unification can be understood as the formal counterpart of unification: while unification computes the most general unifier by making terms equal via appropriate variable assignments, anti-unification constructs more general terms for given terms. From such a generalization process, an analogical relation can be constructed by associating terms with a common generalization.

4.1 First-Order Anti-Unification

First-order anti-unification was introduced by Plotkin [38] in the context of inductive learning. Figure 3 gives several examples for anti-unifications. Terms are generalized resulting in an anti-instance, where differing subterms are replaced by variables [3]. The original terms can be restored by replacing the new variables by appropriate subterms. This idea can be made more precise by introducing the notion of a substitution:

Definition 4 (Substitution) *Assume a term algebra $Term(\Sigma, \mathcal{V})$ is given. A* substitution *on terms is a partial function $\sigma : \mathcal{V} \to Term(\Sigma, \mathcal{V})$ mapping*

[3] Variables introduced by the anti-unification are denoted by capital letters.

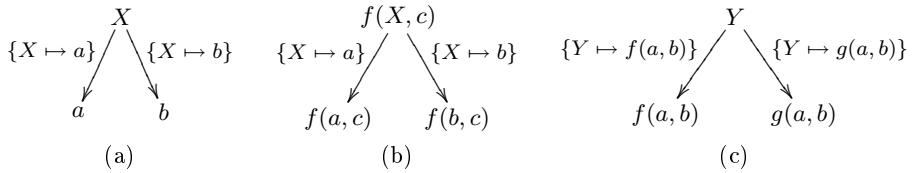


Fig. 3. Plotkin's first-order anti-unification.

variables to terms, formally represented by $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ *(provided* $x_i \neq x_j$ *for* $i, j \in \{1, \ldots, n\}$*,* $i \neq j$ *and sorts of* x_i *and* t_i *match). An application of a substitution* σ *on a term is defined by induction over the term structure:*

- $apply(x, \sigma) = \begin{cases} t & \text{if } x \mapsto t \in \sigma \\ x & \text{otherwise} \end{cases}$
- $apply(f(s_1, \ldots, s_m), \sigma) = f(apply(s_1, \sigma), \ldots, apply(s_m, \sigma))$

We say that a term t' *is an* instance *of* t *and* t *is an* anti-instance *of* t'*, if there is a substitution* σ *such that* $apply(t, \sigma) = t'$*. In this case we write* $t \xrightarrow{\sigma} t'$ *or simply* $t \rightarrow t'$*.*

Using substitutions, generalizations can be defined as follows:

Definition 5 (Generalization) *A generalization for a pair of terms* $\langle s, t \rangle$ *is a triple* $\langle g, \sigma, \tau \rangle$ *with a term* g *and substitutions* σ, τ *such that* $s \xleftarrow{\sigma} g \xrightarrow{\tau} t$*.*

Anti-unification aims to find a most specific anti-unifier, normally referred to as least general generalization (*lgg*), i.e. a generalization that is minimal with respect to the instantiation ordering. [4] It has been proven in [38] that for a given pair of terms a generalization always exists and that the *lgg* is unique (up to renaming of variables).

Figure 3 demonstrates how generalizations can induce an analogical relation: in (a), the terms a and b are generalized to X, therefore b can be seen as an analogon to a in the target domain. In (b), the terms a and b are embedded as function arguments in a common context, but still the same substitutions can be used for generalization and therefore the same analogical relation is established. In (c), the two terms differ with respect to the function symbols. Here, the first-order anti-unification fails to detect the common structure between these terms and generalizes it to X, using rather complex substitutions.

In numerous papers, Gentner and colleagues have shown empirically that analogies are typically characterized by deep structural commonalities (systematicity criterion). For analogy making, Plotkin's first-order anti-unification is not powerful enough, since structural commonalities are ignored if they

[4] This is dual to unification, where a most general unifier (*mgu*) is computed.

are embedded in different contexts. Therefore, HDTP extends classical anti-unification by introducing a new kind of restricted higher-order anti-unification. In the context of analogies, generalizations shall preserve as much of the structure of both domain terms as possible. At the same time, generalizations must not be structurally more complex than the original terms.

4.2 Restricted Higher-Order Anti-Unification

HDTP applies a restricted form of higher-order anti-unification [31] for analogy making. The main problem when dealing with higher-order anti-unification is that generalizations can become arbitrarily complex and may no longer reflect structural commonalities of the original terms. We therefore propose to extend the set of possible generalizations in a controlled way by introducing a new notion of basic substitution. After giving the formal definition, we will illustrate it with an example.

We extend classical first-order terms by introducing variables that can take arguments: for every natural number n we assume an infinite set $\mathcal{V}_n = \{F : s_1 \times \ldots \times s_n \rightarrow s, \ldots\}$ of variables with arity n and $s_1, \ldots, s_n, s \in Sort_\Sigma$. Here we explicitly allow the case $n = 0$ with $\mathcal{V}_0$ being the set of first-order variables. In this setting, a *term* is either a first-order or a higher-order term, i.e. an expression of the form $F(t_1, \ldots, t_n)$ with $F : s_1 \times \ldots \times s_n \rightarrow s \in \mathcal{V}_n$, terms $t_1, \ldots, t_n \in \mathrm{Term}(\Sigma, \mathcal{V})$, and $\mathrm{sort}_\Sigma(t_i) = s_i$. Analogously to the first-order case shown in figure 4, terms can be anti-unified to a generalization subsuming the specific terms. The following list of basic substitutions are applicable in HDTP. These are sufficient for generalizations in analogical reasoning and meet the requirement to only generate less complex anti-instances.

Definition 6 (Basic Substitutions) *We define the following set of basic substitutions:*[5]

(1) A renaming $\rho^{F,F'}$ *replaces a variable* $F \in \mathcal{V}_n$ *by another variable* $F' \in \mathcal{V}_n$ *of the same argument structure:*

$$F(t_1, \ldots, t_n) \xrightarrow{\rho^{F,F'}} F'(t_1, \ldots, t_n).$$

(2) A fixation ϕ_c^F *replaces a variable* $F \in \mathcal{V}_n$ *by a function symbol* $f \in \mathcal{C}_n$ *of the same argument structure:*

$$F(t_1, \ldots, t_n) \xrightarrow{\phi_f^F} f(t_1, \ldots, t_n).$$

[5] To improve readability we omit the sortal specifications of the variable symbols, as long as they can be inferred from the context

(3) An argument insertion $\iota_{G,i}^{F,F'}$ *with* $0 \leq i \leq n$, $F \in \mathcal{V}_n$, $G \in \mathcal{V}_k$ *with* $k \leq n - i$, *and* $F' \in \mathcal{V}_{n-k+1}$ *is defined by*

$$F(t_1, \ldots, t_n) \xrightarrow{\iota_{G,i}^{F,F'}} F'(t_1, \ldots, t_i, G(t_{i+1}, \ldots, t_{i+k}), t_{i+k+1}, \ldots, t_n).$$

(4) A permutation $\pi_\alpha^{F,F'}$ *with* $F, F' \in \mathcal{V}_n$ *and bijective* $\alpha : \{1, \ldots, n\} \to \{1, \ldots, n\}$ *rearranges the arguments of a term:*

$$F(t_1, \ldots, t_n) \xrightarrow{\pi_\alpha^{F,F'}} F'(t_{\alpha(1)}, \ldots, t_{\alpha(n)}).$$

Figure 4 gives examples for all basic substitutions. (a) shows an example for *renaming*: the terms in the source and the target domain both contain variables Y and Z which are generalized to the variable X. Since variables can represent any possible term, it is irrelevant which variable name is chosen. It is possible to align a variable of the source domain with a variable in the target domain without any cost. The renaming substitution is only required for formal reasons: it does not lead to a real generalization.

Argument fixation as shown in (b) can be used to replace a variable in the generalization by a symbol of the same argument structure, e.g. $f(X)$ is replaced by $f(a)$ in the source, respectively $f(b)$ in the target, with $X \in \mathcal{V}_0$. In the Rutherford analogy, this basic substitution is applied in order to map $mass(sun)$ to $mass(nucleus)$ via the generalized term $mass(X)$, fixating X to sun, respectively $nucleus$. Example (c) demonstrates a fixation of a higher-order term $F(a)$ to $f(a)$, respectively $g(a)$. The higher-order variable F has one argument, therefore $F \in \mathcal{V}_1$. In our running example, this substitution is needed to align the gravitation force with the coulomb force.

Argument insertion is a bit more complicated: inserting a 0-ary variable X increases the arity of the embedding term by 1. In (d), a variable X is inserted after the second argument on the source side $F(a, b, c) \to F'(a, b, X, c)$. F' has now four arguments. Inserting a variable $G \in \mathcal{V}_n$ with $n \geq 2$ reduces the arity: On the target side a two-ary variable G is inserted after the first argument $F(a, b, c) \to F''(a, G(b, c))$. F'' has now only two arguments. This basic substitution is required if a complex structure maps on a less complex structure. [6]

An example for *permutation* is shown in (e). Source and target domain contain terms with an equivalent structure. They differ only with respect to the argument order. $F(a, b)$ serves as generalization for both terms. On the target

[6] In the heat flow analogy, which is formalized in appendix A, the height of the water currently being in the vial ($height(in(water, vial), t)$) maps on the temperature of the ice cube ($temp(ice\_cube, t)$). The term in the water flow domain is more complex than the term in the heat flow domain.

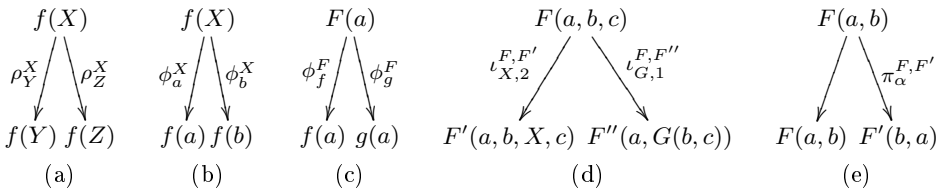


Fig. 4. Examples for all basic substitutions of the restricted higher-order anti-unification.

side, both arguments are permuted and $F(a,b) \rightarrow F'(b,a)$. In our running example, the function $distance(a,b,t)$ is symmetric in the first two arguments. Therefore the order of the arguments in source and target might be exchanged (e.g. $distance(sun, planet, t)$ and $distance(electron, nucleus, t)$). If the anti-unification of other terms proposed that sun maps on nucleus while planet maps on electron, permutation could be applied to change the order of the arguments.

For generalizing complex terms, we can successively apply several substitutions: To receive a non-ambiguous set of substitutions we apply the basic substitutions in the order renaming, argument insertion, permutation, and finally fixation. We will call any composition of basic substitutions a (higher-order) substitution and write $t \rightarrow t'$, if there exists a sequence of basic substitutions that transforms t into t'. Again we will call t' an (higher-order) instance of t, and t an (higher-order) anti-instance of t'.

It has been proven in [31] that the application of a basic substitution will never make a term less complex and so the following fact holds:

Fact 1 *For a given term t there are (up to renaming) only finitely many anti-instances (i.e. terms s with $s \rightarrow t$).*

Fact 1 implies that this notion of substitution is a viable tool to compute generalizations in the context of analogy making. It is a real extension of first-order substitution and, as demonstrated in the examples, it is capable of detecting structural commonalities that are ignored by first-order anti-unification.

4.3 Preferred Generalizations

Replacing the notion of substitution by higher-order substitution, generalizations can be defined exactly like in definition 5. As a direct consequence of fact 1 there exist only finitely many generalizations up to renaming. However, least general generalizations are no longer unique, as demonstrated in figure 5.

Having multiple possible generalizations is not necessarily bad, particularly in the context of analogies, where usually several mappings with different degrees

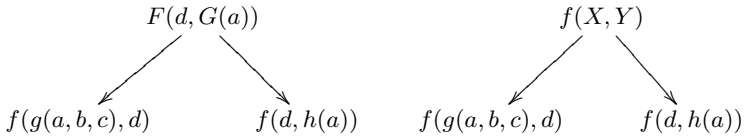


Fig. 5. Example with multiple least general generalizations

of plausibility may coexist. It is generally accepted that often there is not only one analogy between a source and a target, instead several differing solutions are to be expected. However, not all of them might be equally plausible. It would be useful to have a criterion to rank the alternatives. To optimize the analogy making process and compute the preferred analogies, we require a mechanism to evaluate generalizations and select only the preferred ones. The representation of substitutions as compositions of basic substitutions allows for a natural complexity measure:

Definition 7 (Complexity of Substitution) *The* complexity of a basic substitution σ *is defined as*

$$\mathcal{C}(\tau) = \begin{cases} 0 & \text{if } \tau = \rho & \text{(renaming)} \\ 1 & \text{if } \tau = \phi_f & \text{(fixation)} \\ k+1 & \text{if } \tau = \iota_{V,i} \text{ and } V \in \mathcal{V}_k & \text{(argument insertion)} \\ 1 & \text{if } \tau = \pi_\alpha & \text{(permutation)} \end{cases}$$

For a composition of basic substitutions we define $\mathcal{C}(\sigma_1 \ldots \sigma_m) = \sum \mathcal{C}(\sigma_i)$ *and for an arbitrary substitution* σ *the complexity is* $\mathcal{C}(\sigma) = \min\{\mathcal{C}(\sigma_1 \ldots \sigma_m) \mid \sigma_1 dots \sigma_m = \sigma\}$.[7]

The complexity of a substitution is meant to reflect its processing effort. Therefore permutations have a non-zero complexity even though they do not change the structural complexity of a term. The argument insertion restructures the term, and the higher the arity of the inserted variable, the more arguments are moved and therefore the more complexity is assigned to that operation.[8]

Definition 8 (Complexity of Generalization) *Let* $\langle g, \sigma, \tau \rangle$ *be a generalization for a pair of terms* $\langle s, t \rangle$. *Define the complexity of the generalization by* $\mathcal{C}(\langle g, \sigma, \tau \rangle) = \mathcal{C}(\sigma) + \mathcal{C}(\tau)$.

[7] The minimum construction is needed, as there exist multiple decompositions of σ with different complexities. One can define a normal decomposition which can be shown to have minimal complexity. This is left out in this paper due to space limitations.

[8] The complexity values for basic substitutions given in definition 7 have proven to be useful in practice. The analysis of different values is subject of future work.

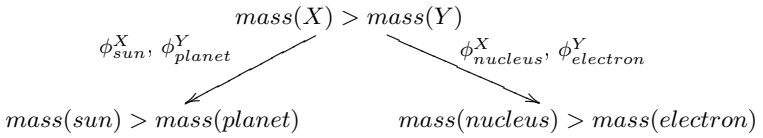
With this complexity measure we can select a *preferred generalization* by minimizing their complexity. Obviously, preferred generalizations are always least general, while the contrary is not always the case as demonstrated in figure 5.

For practical applications it is necessary to anti-unify formulas and not just terms. HDTP extends the notion of generalization from terms to formulas in a straightforward way: from an anti-unification point of view, formulas in clause form can be treated in the same way as terms. This is due to the fact that (positive) literals are structurally equal to function expressions and complex clauses in normal form (e.g. conjunctive normal form) can be processed component wise. Due to space limitations we elaborate the theory only for terms, but it is equally applicable to formulas in clause form and thereby allows the anti-unification of domain axiomatizations as introduced in section 3. [9]

4.4 Reuse of Substitutions

An analogy between two domains is established by anti-unifying the formulas of the target domain with the formulas of the source domain, i.e. not only two formulas, but two sets of formulas are anti-unified. Therefore, the situation might occur in which one formula of the target domain could be aligned with different formulas of the source domain. There may exist different possible and competing anti-unifiers (generalizations).

HDTP uses a sequential heuristic-driven algorithm to compute generalizations: the algorithm consecutively chooses formulas of the target domain and searches for corresponding formulas in the source domain for anti-unification. These generalized formulas together form the generalized theory, which links source and target and represents the commonalities at an abstract level. When choosing the preferred generalization, we have to account for the fact that substitutions which were required earlier to anti-unify two formulas, might be applicable again to the anti-unification of formulas later in the process. It is possible to reuse substitutions without any cost. Therefore the order in which formulas are anti-unified influences the resulting analogy. Assume the following course of events: In our running example, HDTP aligns the formulas $mass(sun) > mass(planet)$ with $mass(nucleus) > mass(electron)$:



[9] In the case of formulas, we consider only admissible substitution, i.e. substitutions that do not introduce variables into the scope of a quantifier.

types
$real$, $object$, $time$
constants
$X : object$, $Y : object$
functions
$mass : object \to real \times \{kg\}$
$dist : object \times object \times time \to real \times \{m\}$
$F : object \times object \times time \to real \times \{N\}$
$centrifugal : object \times object \times time \to real \times \{N\}$
predicates
$revolves\_around : object \times object \times object$
facts
$\gamma_1 : mass(X) > mass(Y)$
$\gamma_2 : mass(Y) > 0$
$\gamma_3 : \forall t : time : F(X,Y,t) > 0$
$\gamma_4 : \forall t : time : dist(X,Y,t) > 0$
laws
$\gamma_5* : \forall t : time,\ o_1 : object,\ o_2 : object :$
$dist(o_1,o_2,t) > 0 \wedge F(o_1,o_2,t) > 0$
$\to centrifugal(o_1,o_2,t) = -F(o_1,o_2,t)$
$\gamma_6* : \forall t : time,\ o_1 : object,\ o_2 : object :$
$0 < mass(o_1) < mass(o_2) \wedge dist(o_1,o_2,t) > 0 \wedge\ centrifugal(o_1,o_2,t) < 0$
$\to revolves\_around(o_1,o_2)$

Fig. 6. Generalization Th_G of the solar system and the Rutherford atom (including the generalizations from the transfer marked with *).

Two fixation substitutions on each side aligning *sun/nucleus* and *planet/electron* were required to compute the generalization. When anti-unifying the next two formulas, e.g. $coulomb(electron, nucleus, t) > 0$ with $gravity(planet, sun, t) > 0$, these substitutions can be reused. It seems plausible that the reuse of already established mappings is cognitively preferred and it is further supported, because it leads to a coherent mapping.

The analogical relation is constructed via successively choosing the preferred generalizations of an axiom from source and target. For the Rutherford analogy, the resulting generalized theory is shown in figure 6. It contains three generalized variables X, Y, and F with the following analogical alignment: $X \to sun/nucleus$, $Y \to planet/electron$, and $F \to gravity/coulomb$.

4.5 Advantages versus Disadvantages of Anti-Unification

HDTP successively generalizes two aligned formulas from source and target and constructs a generalized theory which explicitly describes the commonalities at an abstract level. We consider this as one of the main advantages of HDTP, because it not only allows for analogical reasoning in the target domain, but also for analogical generalization, which is essential for human learning via abstraction (compare section 7).

Applying the theory of restricted higher-order anti-unification to establish an analogical relation between formulas is very powerful: unlike many other analogy models, HDTP matches functions and predicates with same and dif-

ferent labels, but anti-instances for identical function and predicate names are more specific than for differing ones. The same holds for structural differences: HDTP also matches formulas with different structure, which might end up in a very generic anti-unifier. The substitution complexity is used to guide this anti-unification and lead to an overall meaningful generalization. HDTP supports the idea of one-to-one mapping in the heuristic, but does not reject alignable formulas which violate the one-to-one mapping criterion. The major restriction of HDTP alignment is logical consistency: an alignment is only accepted if the generalized formula is consistent with the rest of the generalized theory. Additionally, to avoid nonsensical alignments and to compensate the weak restrictions of higher-order anti-unification, HDTP introduces other constraints such as including sortal information in the mapping: only entities of the same sort may be aligned.

The alignment process in HDTP is sequential and requires decisions on how to proceed: the algorithm has to choose the next term to anti-unify and it has to choose the preferred anti-instance. This process is guided via heuristics: the order in which formulas are chosen to find an alignable match is determined by the structural complexity and starts with the simplest ones. Another heuristic determines the generalization to a least general anti-instance. Those generalizations are preferred that conserve as much information as possible, i.e. those generalizations are preferred that maximize the analogical relation and the generalized structure, because every substitution implies information loss. Moreover, the substitutions required to transform a generalized to a specific formula should be as simple as possible, since the complexity of substitutions indicates the degree of structural divergence. Second-order substitutions are to be avoided.

5 Re-Representation

Analogy making is based on a mapping of individuals and formulas between two domains. While the creation of such an analogical mapping is well examined, if the corresponding representations of both domains are already chosen in a way that the structural compatibility is obvious, such situations seem to be somewhat artificial. In fact, the structural commonalities characterizing two analogous domains are usually not obvious in advance, but become visible as a result of the analogy making process. The conceptualization (i.e. the representation) must be modified and adapted to make implicit analogous structures explicit. It is argued that an essential part of establishing an analogy is a change of representation of one or both domains to allow for discovering the common structure [25].

Although analogical reasoning and the development of computational models

for analogy making has a relatively long history of more than thirty years, it happened to be rather recently that research on re-representation has been recognized as an important aspect of analogy making. Most prominently Indurkhya's work [25] should be mentioned as a foundation of re-representation. He develops a theory where the computation of analogies is based on the *accommodation* of an internal concept network to an input (resulting in a re-representation of the concept network), or the *projection* of a concept network to the input (resulting in a re-representation of the input), or both. Re-representation techniques were also applied to geometric figures [35] and the string domain [20], two classical domains where structural descriptions of objects need to be rerepresented in order to compute analogical relations. Although only little attention was paid to re-representation in the structure-mapping tradition at the beginning [6], this changed due to the application of structure-mapping to real-world problems, such as physics problems taken from textbooks: in [42], a theory of re-representation in the SME tradition is presented based on operations such as transformation, decomposition, or entity splitting.

In this section, we show that the logical framework used in HDTP entails a mechanism for re-representation in a quite natural way: a logical representation of a domain does not only provide the axioms, that are explicitly given, but also makes all formulas available that can be inferred from the axioms by logical deduction. The idea is to incorporate these derived formulas into the mapping process, if the original representation given by the axioms does not lead to a satisfying analogical relation. We will exemplify this idea by our running example, and then give a formal treatment of this re-representation mechanism.

5.1 Re-Representation Exemplified

The original axiomatization of the Rutherford analogy in figure 2 was chosen in a way, such that the analogy could directly be discovered by matching the axioms of the two domains one by one. In figure 7, we give a different axiomatization of the two domains. We do not explicitly state that *sun* and *planet* as well as *nucleus* and *electron* attract each other, but we formulate general laws from which these facts can be derived. Notice that this new formalization is more general, as all axioms of figure 2 can be inferred from those of figure 7.

Given this new representation of the Rutherford analogy, there is no way to achieve a generalized formula $\forall t : attracts(X, Y, t)$ by anti-unifying axioms, since neither the axiomatization of the source domain, nor that of the target domain contains an axiom that would be an instance of this formula. However,

Solar System		Rutherford Atom	
α_1 :	$mass(sun) > mass(planet)$	β_1 :	$mass(nucleus) > mass(electron)$
α_2 :	$mass(planet) > 0$	β_2 :	$\forall t : distance(electron, nucleus, t) > 0$
α_3 :	$\forall t : distance(sun, planet, t) > 0$	β_3 :	$charge(nucleus) > 0$
α_4 :	$\forall x \forall y \forall t : mass(x) > 0 \wedge mass(y) > 0 \rightarrow gravity(x, y, t) > 0$	β_4 :	$charge(electron) < 0$
α_5 :	$\forall x \forall y \forall t : gravity(x, y, t) > 0 \rightarrow attracts(x, y, t)$	β_5 :	$\forall x \forall y \forall t : charge(x) > 0 \wedge charge(y) < 0 \rightarrow coulomb(x, y, t) > 0$
α_6 :	$\forall x \forall y \forall t : attracts(x, y, t) \wedge distance(x, y, t) > 0 \wedge mass(x) > mass(y) \rightarrow revolves\_around(y, x)$	β_6 :	$\forall x \forall y \forall t : coulomb(x, y, t) > 0 \rightarrow attracts(x, y, t)$
Background Knowledge			
ϕ_1 :	$\forall x \forall y \forall t : distance(x, y, t) = distance(y, x, t)$		
ϕ_2 :	$\forall x \forall y \forall z : x > y \wedge y > z \rightarrow x > z$		

Fig. 7. Another Formalization of the Rutherford analogy

using logical deduction, the formula

$$\forall t : gravity(sun, planet, t) > 0$$

can be derived from $\{\phi_1, \alpha_1, \alpha_2, \alpha_4\}$ on the source side, which can be anti-unified with

$$\forall t : coulomb(nucleus, electron, t) > 0$$

which in turn can be inferred from $\{\beta_3, \beta_4, \beta_5\}$ on the target side.

By allowing the application of logical deductions prior to anti-unification, we also have another way to address the problem of argument swap concerning the *distance* function, which has been discussed in section 4.2. Notice that the parameters of β_2 are switched compared to the formalization in figure 2, so that anti-unifying α_2 and β_2 would result in the unwanted mapping of *sun* to *electron* and *planet* to *nucleus*, which contradicts the mapping established by anti-unifying α_1 and β_1. In this situation, another representation of this axiom would exhibit the common structure of the two domains in a way such that anti-unification leads to an appropriate generalized theory: from background knowledge it is known that *distance* is a symmetric function, i.e.

$$\forall t : distance(nucleus, electron, t) > 0$$

can be derived from $\{\phi_1, \beta_2\}$, which is a good candidate for anti-unification with α_3, as it allows the reuse of substitutions.

In general, the task of re-representation consists of finding pairs of formulas from the domain theories, that posses a common structure and therefore lead to a good generalization. In this example, re-representation is just needed to enhance the support for the analogy, i.e. to increase the number of formulas that can be anti-unified with the same set of substitutions. However, cases

exist in which no useful analogy can be computed at all, if only the given axiomatizations for the two domains is considered.

5.2 Formal Treatment

As usual, we will write $Ax \vdash \phi$ if formula ϕ can be derived from a set of axioms Ax and we denote the set of all formulas that can be derived from Ax by $Th(Ax)$. We can state the idea behind the method of re-representation as follows: do not only consider the formulas given by the axiomatizations Ax_S and Ax_T of the source and the target domain for anti-unification, but all formulas from the theories $Th(Ax_S)$ and $Th(Ax_T)$. Therefore we extend the notion of anti-unification to theories:

Definition 9 *Let G be a finite set of formulas.*

(1) We call G an anti-instance *of a set of formulas F iff there exists a substitution σ such that $Th(apply(G,\sigma)) \subseteq Th(F)$. Again we will write $G \xrightarrow{\sigma} F$ or just $G \to F$.*

(2) Let Ax_S, Ax_T be sets of formulas and σ, τ substitutions. We call the triple $\langle G, \sigma, \tau \rangle$ a generalization *of Ax_S and Ax_T iff $Ax_S \xleftarrow{\sigma} G \xrightarrow{\tau} Ax_T$.*

Notice, that we only do inference on the set $apply(G,\sigma)$ but not in G directly. Furthermore, we do not require that there exist anti-instances for all formulas of F in G. Hence the empty set is an anti-instance of every set of formulas, and also a generalization for all pairs of sets of formulas. However, this is probably not a generalization we are looking for, since it results in the empty analogical relation. Therefore, we introduce the concept of coverage:

Definition 10 *Given a generalization $\langle G, \sigma, \tau \rangle$ of Ax_S and Ax_T, the subset $Th(apply(G,\sigma))$ of $Th(Ax_S)$ is said to be* covered *by G and for Ax_T accordingly.*

Obviously, adding formulas to G will never decrease the coverage. More generally one can state:

Fact 2 *An anti-unifier $\langle G, \sigma, \tau \rangle$ has at least the same coverage as $\langle G', \sigma', \tau' \rangle$ if there exists a substitution $G' \xrightarrow{\theta} G$ that is compatible with the domain substitutions (i.e. $\sigma' = \sigma \circ \theta$ and $\tau' = \tau \circ \theta$).*

In general, a greater coverage is preferable, since it provides more support for the analogy. However, there are some caveats: One can construct examples where it is possible to extend the coverage of the generalization by adding an infinite number of formulas, that can be derived from formulas that are not covered by the generalization. This can be done without changing the substitutions, and therefore no additional knowledge about the analogical relation

is gained by extending the generalization in this way. As seen in the example in section 5.1, some axioms are not covered by the generalization, while they nevertheless have been used to derive formulas that can be generalized. We will call these axioms *indirectly covered* by the generalization. The computation of the analogical relation can be stopped if all axioms for a given domain are indirectly covered.

In many cases, it is possible to extend the coverage by creating double mappings. In the Rutherford analogy, one might extend the generalization by adding $\langle\{H(Z) > 0\}, \sigma, \tau\rangle$ with $\sigma = \{H \mapsto \mathit{mass}, Z \mapsto \mathit{planet}\}$ and $\tau = \{H \mapsto \mathit{charge}, Z \mapsto \mathit{nucleus}\}$. However, with this additional formula not only coverage is increased, but now the analogical relation maps *mass* to *mass* as well as *mass* to *charge*, a situation that is normally undesirable. A good strategy to compute an analogical relation is to maximize the (indirect) coverage of the domains while avoiding double mappings.

5.3 Discussion

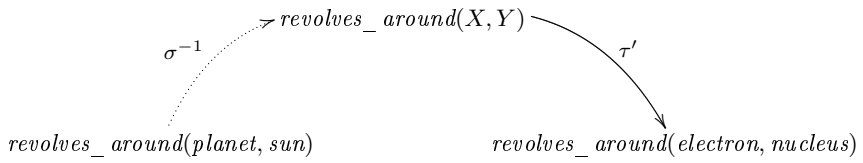
Re-representation is a major challenge in computing analogies. To allow for automatic re-representation, the representation mechanism applied has to offer a notion of equivalence of different formalizations, and should provide means to create new representations for a given formalization. Here, the choice of logic as a representation mechanism exhibits its power, since beside the formulas explicitly given by an axiomatization there are also implicit formulas that can be inferred from these axioms. This provides an integrated notion of representation, where other analogy-models have to introduce special and sometimes quite artificial means.

However, there are also problems associated with this approach: the number of formulas that can be derived is not finite and therefore strategies have to be developed to control the re-representation process. The integration of mapping and re-representation seems to be necessary, as only the comparison of source and target formalization can exhibit information that are needed to guide re-representation. Future work concerns a theoretical assessment of the trade-off between the maximization of coverage and the minimization of substitution complexities. These considerations lead to the formulation of heuristics and an algorithmic treatment of re-representation which allows for a thorough practical evaluation of the proposed approach.

6 Analogical Transfer

Although analogical mapping is the central point of concern in most analogy models and the establishment of an analogical relation takes most of the space in literature on analogy, the mapping phase is usually not the goal, but just a step to prepare an analogical transfer. During this transfer, knowledge from the source domain is translated using the analogical mapping, to hypothesize new statements about the target domain. The kind of knowledge transferred depends on the context in which the the analogy is established: in analogical reasoning tasks just one statement, the analogical inference, is transferred. In analogical problem solving, analogical theorem proving, or analogical modeling more complex structures might be transferred.

In the framework of HDTP, the analogical mapping is computed via a generalization process: matching terms of the source and target domain are generalized to a common variable in the generalized theory. Thereby a correspondence of terms of the source and target domain is established, which can be used to translate formulas of the source domain into the target domain. Although there is no general way to predict which formulas of the source domain are good candidates for the analogical transfer, it seems to be plausible to propose formulas that are not already covered by the generalized theory (in the sense discussed in section 5.2), as these may induce new knowledge about the target domain. However, it should be kept in mind, that although these analogical inferences are produced within a logical framework, they are not logical inferences in the classical sense, i.e. they can create new and even contradicting statements. HDTP can detect inconsistent inferences by applying a standard theorem prover. However, there is no generally applicable method to judge the usefulness of an analogical inference, as this varies with the given task. For example, in analogical problem solving, the utility of an analogical inference is determined by its contribution to a solution of the problem.



In the context of HDTP, two types of analogical transfers are distinguished: first, there are new statements about the target domain, that can be formulated within the existing vocabulary. This is exemplified in our running example: the statement *revolves_ around*(*planet*, *sun*), which can be inferred from the source domain, can be projected to *revolves_ around*(*electron*, *nucleus*) in the target domain. This analogical inference does not follow from the axiomatization of the target domain and therefore provides new (hypothetical) knowl-

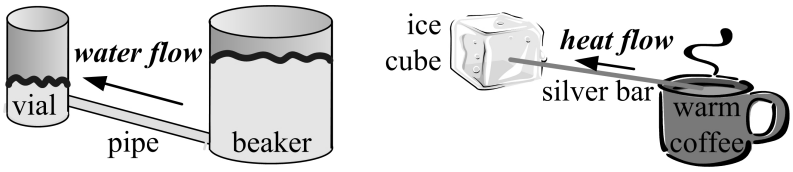


Fig. 8. Heat Flow Analogy

edge. It is acceptable from a logical point of view, since it does not contradict the existing axioms. It is therefore reasonable to consider the hypothesis and further investigate, whether the knowledge should be incorporated into the axiomatization of the target domain. How this decision is made, depends on the kind of knowledge that is represented. In the case of our running example, in which a model of a physical domain is constructed, physical experiments are an appropriate means to assess the quality of the analogical inference: in fact, Rutherford's atom model was soon superseded by Bohr's atom model, due to some early quantum mechanical results.

Another kind of analogical transfer introduces entirely new concepts into the target domain. A classical example for this kind of concept creation by analogical transfer is provided by the heat flow analogy. In this analogy, two connected vessels filled with different quantities of water are analogically related to two massive bodies of different temperature that are connected via some metal bar. Although the height of the water in the vessels can be related to the temperature of the objects, and change of temperature corresponds to change of water level, there is a priori no direct correspondence to the water in the vessels. However, such a correspondence can be created by introducing a new concept, usually referred to as *heat*, whose properties are induced by analogical transfer. This new concept, although not directly observable in a physical sense, allows to model and predict the observable change of temperature. In the context of HDTP, such concept creations occur, when formulas are proposed for transfer, that contain subterms which are not mapped to terms of the target domain. We give a formal treatment of the heatflow analogy in appendix A.

If an analogical inference has been accepted, the target theory can be extended by the new formula. As the inference corresponds to knowledge from the source domain, it increases the coverage of the analogical mapping and broadens the basis of the analogy. In the framework of HDTP, this implies that the generalized theory can also be extended by the transferred formula, reflecting a greater structural commonality of the two domains.

7 Modeling Cognitive Abilities with HDTP

Analogies are central to the understanding and modeling of cognitive phenomena and play an important role in many different cognitive abilities [16]. The previous chapters have reviewed HDTP primarily under a computational perspective and explained the formal mechanisms applied in analogy making. In this chapter, we compare HDTP to other, cognitively inspired or cognitively plausible analogy models. We discuss several characteristics of analogies and their role in cognitive abilities.

Although there has been a long tradition of research on analogical reasoning methods in artificial intelligence and cognitive science, there do not exist many formal theories on analogical inferences that could be compared to classical inference mechanisms such as induction, deduction and abduction. HDTP extends these reasoning mechanisms to analogical reasoning, a reasoning mechanism commonly applied as source of knowledge in creative inventions [34] and human learning [11]. HDTP is a formal model based on a logical language in the tradition of artificial intelligence. Logical languages have proven suitable to model knowledge on domains, however they have been criticized for being restricted to logical inferences. It is unclear to what extent logical reasoning can be called cognitively plausible.

There is widespread agreement that analogy is based on an alignment of relational or structural commonalities between a source and a target domain [21,10,6]. Analogies typically rely on large and deeply interconnected systems of relations that hold among elements from the source domain and among elements of the target domain as well. HDTP follows the same strategy: It uses anti-unification, a syntactical operation to compare formulas at a structural level, and detects structural correspondences between two logical theories. By re-using existing substitutions, i.e. by applying a mapping that has already been used before, a consistent and coherent overall-mapping is established, similarly to the systematicity principle in SME [10].

Like most approaches to model analogy making [11,25,19,23,24], HDTP prefers a one-to-one mapping. However, it does not reject multiple alignments per se. A one-to-many or many-to-many mapping is often an indication of different possible conceptualizations of the domains. The representation must be adapted to make the analogous structures obvious (e.g. [7]). In section A.3 we show an analogy with many-to-many mappings.

Analogy provides new insight about the target domain, but it is also fundamental in learning general principles [2] and abstract concepts or categories. While classical inductive learning requires a large set of data samples to create general laws, humans can generalize already over a small set of samples by

applying analogical comparison. In [13,32,2], Gentner et al. showed that relational categories are better learned by a common abstract relational structure or rules rather than common properties. Reflecting this analogical generalization process is one of the strengths of HDTP: during the analogical mapping, the anti-unification automatically constructs a generalization for every aligned pair of terms. This way, HDTP creates an explicit generalized theory over two domains – the source and the target domain. This general theory contains common structures at an abstract level, but also common rules. In [15], is shown that it is possible to successively refine this generalized theory via analogical comparison to other domains.

An analogy between two different domains is established when analogous entities and relations are aligned and mapped to each other. Often, analogy models assume that representations of source and target domain are already available in a way that the structural commonalities are obvious. We consider the process of finding a suitable representation as part of analogy making. In [26], it is argued that commonalities are not there in advance, but are created in the analogy-making process. Detecting analogous structures is strongly connected with seeing things in a new and different way. In fact, humans handle incompatible information every day. We argue, that an essential part of analogy making is the change of representation of one or both domains to allow for discovering the common structure: establishing an analogy between two different domains means transforming two, in the first place incompatible conceptualizations to align analogous elements and establish a compatible, analogous relation between them. HDTP actively supports this process of re-formulation of knowledge: based on a deductive reasoner it is possible to re-represent given formalizations of source and target to a suitable and compatible representation. This may lead to a deeper understanding of already well established domains.

HDTP is one integrated framework and processes the different tasks – analogical mapping, re-representation and generalization – in an integrated way. This is in contrast to other analogy models such as SME, which separates these different tasks and models them in different modules. In accordance to Chalmers, French and Hofstadter [1], we argue that analogy making is complex, high-level cognition which should not be seen as sequence of separate, but as one integrated and interacting cognitive process.

Creativity is a cognitive ability which plays an important role in human reasoning and problem solving. Creativity as defined in [3, p. 205] is "the production of an idea, action, or object that is new and valued". Analogies can provide a means for creativity, as they can introduce new knowledge into a domain via analogical transfer. The principles guiding the transfer ensure that the concepts are relevant in order to understand the target domain. HDTP models creativity in different ways. First, analogical transfer is used to propose new

entities in the target domain, to generate new knowledge about existing entities, but also to hypothesize new concepts which would not have been possible to conceptualize without the analogy. For instance in the heat flow analogy, "heat" in contrast to temperature is not observable and can be conceptualized only via analogy to a perceivable domain, e.g. water flowing between two vessels with different water levels. The height of the water is aligned with the temperature. From the observation that water keeps flowing until it has the same height in both vessels (source domain) and that temperature tends towards an equilibrium and balances after some time (target domain), it can be inferred via analogy that there must exist an analogous "flowing thing" on the target side: the concept heat. Creative generation of knowledge of this type is classically modeled via abduction[5]. Using analogy in this abductive process guides and motivates why certain things are hypothesized and others not. Second, HDTP models creativity via analogical generalization. Establishing an analogy between two domains also leads to the construction of new concepts at a general level: e.g. in the Rutherford analogy the sun and the nucleus, respectively the planet and the electron are aligned. For the purpose of the analogy they form the ad-hoc concept "central body" and "orbiting object" respectively. These ad-hoc concepts can emerge as permanent concepts after several learning cycles.

8 Related Work

Various approaches to model analogy making have been developed. We review several well-known symbolic and hybrid approaches and outline the commonalities and differences to HDTP.

Probably the best-known theory of analogy making is the Structure Mapping Theory (SMT) [10] and its implementation simulating the structure mapping process, the Structure Mapping Engine (SME) [6]. SME uses graph structures to describe the entities in a domain, their attributes, functions, and the relations between entities. In this approach, the core of the analogy making process is the structure-mapping process. It is complemented by pre- and postmodules for additional tasks such as reasoning and re-representation of knowledge. The representation formalism is not restricted and might be of the same expressivity as HDTP, but the core structure mapping is graph matching, i.e. the mapping does not account for the semantics of for example quantified variables in general laws as HDTP does. Structural commonalities between a source and a target domain indicate an analogy: SME follows the systematicity principle and supports analogies which comprise a hierarchically deep match, because this indicates a structural system of interconnected knowledge. The systematicity principle states that an element belonging "to a mappable system of mutually interconnecting relationships is more likely to be imported

into the target than an isolated predicate" [10, p. 163]. SME identifies structural commonalities via graph-matching techniques in a stepwise process: first, it searches for identical relations in source and target to create local match hypotheses. Match hypotheses are also created for arguments of matching relations if they are both entities or both functions. These local match hypotheses are evaluated according to local evidence scores which also include systematicity. This is a parallel process. Afterwards, the global match construction composes gradually larger and consistent mappings to construct the best inter-domain mapping. While SME aligns entities and functions without matching literally, the alignment of attributes and relations requires identical labels. Attributes are only mapped on attributes, functions on functions, predicates on predicates and they have to have the same arity. The mapping in HDTP is less rigid than in SME: HDTP aligns any entity, function or predicate, however it prefers literally-matching alignments to non-literally alignments and same structures to structural mismatches. Since often knowledge can be modeled equally well as attributes, functions or predicates [1] and this design decision is up to the knowledge engineer, it seems to be reasonable to relax the mapping restrictions. The heuristic used in the overall mapping process is sequential; a parallel heuristic could be applied as well, but is computationally very expensive. A further difference between HDTP and SME results from the way a mapping between two domains is established: while HDTP constructs an explicit generalization, SME's mapping is constructed without generalizing across domains. A generalization is constructed only on demand [32]. SME has been applied to various domains such as strategy games [18], naive physics [29] and analogies in natural-language [8], which we consider as typical domains for HDTP as well.

Indurkhya et al. [25,27] developed another influential framework for analogy making: the Interactionist Theory uses classical algebras to represent source and target domain by $\mathcal{S} = \langle A, \Sigma \rangle$ and $\mathcal{T} = \langle B, \Omega \rangle$ where A and B are sets and Ω and Σ are sets of operations defined on A and B. The analogical (or metaphorical) relation $\langle R, \Psi \rangle$ is considered again as an algebra. Indurkhya's framework and its successors (e.g. [4]) model analogies nicely in the string and the geometric domain, in particular proportional analogies of the form A : B :: C : ? (read: A is to B as C is to what?). Notice that in examples of proportional analogies usually it is assumed that the source objects A and B are taken from the same domain as the target objects C and ?. However, it is not easy to see how Indurkhya-style frameworks can be applied to domains which cannot be represented in such a straightforward algebraic way. For example, how can this framework be used to model analogies that require full first-order logic for a description of the source and target domains?

Copycat is a "computer program designed to discover insightful analogies, and to do so in a psychologically realistic way" [19, p. 205]. Hofstadter and Mitchell developed Copycat at a general level of an architecture, which computes not

only analogies but aims to model human cognition, more precisely to model the development of fluid concepts. Copycat is a hybrid of connectionist and symbolic analogy models and consists of three major components: Slipnet is the long term memory containing a network of concepts, Workspace represents the working memory, which contains instances of concepts. In the string domain, the Workspace contains source and target strings such as "abc" and "abd" . These strings have structured representations, e.g. a successor relation between "a" and "b". The Coderack which contains a framework of suggestions of competing or cooperating relations between instances of the workspace. Copycat's domain is hard-wired into it. While HDTP is an open domain system, Copycat is limited to string analogies. However, like HDTP, Copycat can re-represent, i.e. adapt the structured representation of the domains.

Kokinov [30,36,37] proposes DUAL, a hybrid cognitive architecture integrating the connectionist and symbolic approaches. Cognitive processes like analogical reasoning and the estimation of semantic similarity play an important role in DUAL. The system consists of a network, the DUAL memory, representing the system's world knowledge. The nodes of the network, called micro-agents, have an inner structure and are in a state affecting their behavior in cognitive processes. Such processes are triggered by activation potential and marker passing along the connections between the micro-agents. An initial activation pattern is determined by context and then propagated according to the characters of the individual agents and their interconnections. The activation pattern represents the contents of the short-term memory, while the whole micro-agent network corresponds to long-term memory. The relation between source and target concepts are visualized via spreading activation from source and target through the network. For example, the framework has been demonstrated to be capable of analogical problem solving: given the task to heat water in a wooden vessel in a forest with only matches and a pocket knife the system proposed, analogous to an immersion heater, to warm up the knife and then put it into the water. Since DUAL is a hybrid analogy model, its representation of domains and the way analogical relations are established differ very much from HDTP. Specifically, it is unclear how generalization can be handled in this framework.

Other neuro-inspired analogy models such as LISA [22], ARCS [41], VSA [9] are not explicitly compared to HDTP, as the way of representation and establishing an analogical relations differ significantly.

9 Summary and Future Work

This paper gives an overview of the syntactic principles of Heuristic-Driven Theory Projection (HDTP), a framework to analyze the analogical relation

between a source and a target domain. HDTP is a syntactic analogy model, which describes source and target domain with first-order logic theories. A restricted form of higher-order anti-unification is used to determine structural commonalities between the source and the target at a syntactic level and compute a generalization with substitutions for both domains. The analogical relation is established via the generalized theory.

We argue for using a logical system to represent knowledge, because it is possible to automatically infer knowledge. A deductive reasoner can be used to re-represent knowledge, i.e. deduce new formulas from the given set of target domain axioms, which are logically inferable but match the source domain better in a structural manner. We believe that analogy making essentially relies on the ability of re-representing knowledge exactly in such a way, so that the structural commonalities become visible and the analogy can be established. HDTP provides an inherent mechanism for such a re-representation.

Additional knowledge about the source domain can be transferred to the target domain using the generalization with its substitutions established by the analogical relation. The analogical transfer has only hypothetical status in the target domain: Logical consistency may serve as first indicator whether the analogical transfer is correct. In particular in the physics domain, experiments should be created to systematically evaluate the new knowledge.

HDTP can be used to explain different cognitive phenomena and human learning. For example, the creative generation of new knowledge can be modeled via the analogical transfer. The analogy guides and motivates why certain new facts or laws are hypothesized and others are not. Analogical comparison may also lead to a formation of new, abstract concepts (such as the central-force-system with its central body and satellite in the Rutherford analogy). This generalization process is possible, because different, at the first glance incompatible domains, become comparable via establishing the analogy. Therefore, HDTP models analogical learning via the analogical transfer (like most other analogy models), but it models the analogical generalization process via the construction of the generalized theory [15].

This paper focuses only on the syntactic principles of HDTP. However, one advantage of our formal logic analogy model is the explicit separation of syntax and semantics. Semantic aspects of the analogical relation in HDTP have been discussed in [17] and [14].

The HDTP framework has been implemented in PROLOG, currently however it comprises only a restricted implementation of re-representation. The completion of this framework is still ongoing. Heuristics play an important role in controlling the efficiency of the analogy making process: according to the heuristics applied, HDTP starts with a target domain, orders the axioms

according to their complexity. During the anti-unification process, the complexity of the required substitutions are minimized. Further heuristics need to be developed and evaluated against other analogy models and in psychological experiments.

In future work, we will develop several extensions of our analogy model. As HDTP uses a formal logic knowledge representation in the tradition of classical AI, it is suitable for an integration with knowledge bases such as Cyc. A suitable retrieval mechanism is required to select knowledge about a relevant source domain for a given target problem. Further work will investigate the role of analogical reasoning in an overall context of human learning. HDTP will function as one reasoning unit in the cognitive architecture I-Cog [33].

Acknowledgement

The work was supported by the German Research Foundation (DFG) through the project "Modeling of predictive analogies by Heuristic-Driven Theory Projection" (grant KU 1949/2-1).

A Appendix

This section shows formalizations of the source and target domain for different analogies and gives a generalization and the corresponding mappings.

A.1 Heat Flow Analogy

As argued before, analogies provide a means to understand abstract concepts by relating them to observable phenomena. A classical example is the heat-flow analogy which has been introduced informally in section 6. We will now give a detailed description of this analogy within HDTP. We choose a formalization that includes the following pieces of knowledge:

- In the source domain, there are two vessels, a *beaker* and a *vial*, that are connected via a *pipe* (α_1).
- The vessels are filled with water and initially the height of the water in the beaker is higher than the height of the water in the vial.
- There is water flowing from the beaker to the vial which can be observed as the height of the water in the beaker decreases (α_8) while the height of the water in the vial increases.

- The total amount of water in the two vessels does not change (α_9).
- In the target domain, the coffee in the cup is connected to the ice cube via a sliver bar (β_1).
- It is observable that the temperature of the coffee in the cup decreases while the ice cube gets warmer (β_3).

It should be noted, that water and coffee are mass terms that do not refer to any specific object in the domains and therefore can not be measured by functions like *height* or *volume*. We introduce the operator *in* that denotes a certain amount of liquid inside some container, so that temperature, height, and volume can be assigned to it.

Water Flow (Source)

---

sorts
$real,\ massterm, object,\ time$
entities
$vial : object,\ beaker : object,$
$water : massterm,\ pipe : object,\ t_{start} : time$

functions
$height : object \times time \rightarrow real \times \{cm\}$
$footprint : object \times time \rightarrow real \times \{cm^2\}$
$in : object \times massterm \rightarrow object$
$volume : object \times time \rightarrow real \times \{cm^3\}$
predicates
$connected : object \times object \times object$

facts
$\alpha_1 : connected(beaker, vial, pipe)$
$\alpha_2 : \forall t_1 : time, t_2 : time : footprint(beaker, t_1) = footprint(beaker, t_2)$
$\alpha_3 : \forall t_1 : time, t_2 : time : footprint(vial, t_1) = footprint(vial, t_2)$
$\alpha_4 : \forall t : time : volume(in(water, beaker), t) = footprint(beaker, t) * height(in(water, beaker), t)$
$\alpha_5 : \forall t : time : volume(in(water, vial), t) = footprint(vial, t) * height(in(water, vial), t)$
$\alpha_6 : \forall t : time : footprint(beaker, t) > footprint(vial, t)$
$\alpha_7 : height(in(water, beaker), t_{start}) > height(in(water, vial), t_{start})$
laws
$\alpha_8 : \forall t_1, t_2 : time :$
$t_2 > t_1 \wedge connected(beaker, vial, pipe)$
$\wedge\ height(in(water, beaker), t_1) > height(in(water, vial), t_1)$
$\rightarrow height(in(water, beaker), t_1) > height(in(water, beaker), t_2)$
$\alpha_9 : \forall t_1, t_2 : time,\ o_1, o_2, o_3 : object :$
$t_2 > t_1 \wedge connected(o_1, o_2, o_3)$
$\rightarrow volume(in(water, o_1), t_1) + volume(in(water, o_2), t_1)$
$= volume(in(water, o_1), t_2) + volume(in(water, o_2), t_2)$

Heat Flow (Target)

---

sorts
$real,\ massterm,\ object,\ time$
entities
$coffee : massterm,\ ice\_cube : object,$
$cup : object,\ bar : object,\ t_{start} : time$

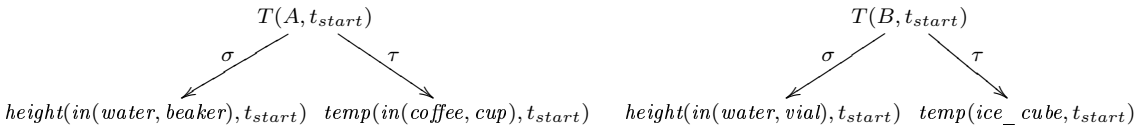
functions
$temp : object \times time \rightarrow real \times \{C\}$
$in : object \times massterm \rightarrow object$
predicates
$connected : object \times object \times object$

facts
$\beta_1 : connected(in(coffee, cup), ice\_cube, bar)$
$\beta_2 : temp(in(coffee, cup), t_{start}) > temp(ice\_cube, t_{start})$
laws
$\beta_3 : \forall t_1, t_2 : time :$
$t_2 > t_1 \wedge connected(in(coffee, cup), ice\_cube, bar)$
$\wedge\ temp(in(coffee, cup), t_1) > temp(ice\_cube, t_1)$
$\rightarrow temp(in(coffee, cup), t_1) > temp(in(coffee, cup), t_2)$

When computing the generalized theory, comparison of the terms

$$height(in(water, beaker), t_{start}) \quad \text{and} \quad temp(in(coffee, cup), t_{start})$$

might lead to the assumption, that *water* should be mapped to *coffee* and *beaker* to *cup*, an assignment that testifies a fundamental misunderstanding of the analogy. However, considering the other parts of the axiomatizations, a preferred generalization in the sense of section 4 will produce the following mapping:



Here the following substitutions are applied:

$$\sigma = \{\ T \mapsto \lambda x \lambda t.height(in(water, x), t), \quad A \mapsto beaker, B \mapsto vial\}$$

$$\tau = \{\ T \mapsto \lambda x \lambda t.temp(x, t), \quad A \mapsto in(coffee,cup), B \mapsto ice\_cube\}$$

Notice that all these substitutions can be expressed as compositions of the basic substitutions introduced in section 4.2. For example the substitution $T \mapsto \lambda x \lambda t.height(in(water, x), t)$ is the following sequence of basic substitutions:

$$\phi^{T'}_{height} \circ \phi^{I'}_{in} \circ \phi^{X}_{water} \circ \iota^{I,I'}_{X,0} \circ \iota^{T,T'}_{I,0}$$

The generalized theory will comprise the following axioms:

Generalization

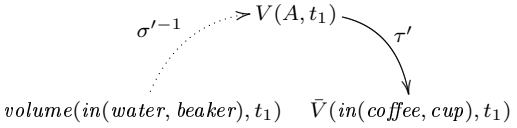
---

sorts
real, *massterm*, *object*, *time*
entities
A : *object*, B : *object*, C : *object*,
t_{start} : *time*

functions
$T : object \times time \to real \times \{Unit\}$
$in : object \times massterm \to object$
predicates
$connected : object \times object \times object$

facts
$\gamma_1 : connected(A, B, C)$
$\gamma_2 : T(A, t_{start}) > T(B, t_{start})$
laws
$\gamma_3 : \forall t_1, t_2 : time :$
$t_2 > t_1 \wedge connected(A, B, C) \wedge T(A, t_1) > T(B, t_1)$
$\to T(A, t_1) > T(A, t_2)$

When computing hypotheses for the analogical transfer, unused parts of the source axiomatization are considered. The term $volume(in(water, beaker), t_1)$ contains the symbol *volume* for which no generalization exists. However, the rest of the term can be generalized using existing substitutions, introducing a variable for the new symbol. This leads to the following proposal for transfer:



with the extended substitution

$$\sigma' = \sigma \cup \{V \mapsto \lambda x \lambda t.volume(in(water, x), t)\}$$

The new substitution can be expressed as

$$\phi^{V'}_{volume} \circ \phi^{I'}_{in} \circ \phi^{X}_{water} \circ \iota^{I,I'}_{X,0} \circ \iota^{V,V'}_{I,0}$$

so it reuses the basic substitutions $\phi^{I'}_{in}$, ϕ^{X}_{water} and $\iota^{I,I'}_{X,0}$ and therefore fits well into the analogy. On the target side, the projection of the new Variable V, which is denoted by $\bar{V}$, lacks an interpretation and thereby indicates that some new concept needs to be introduced to understand the situation, in this case the concept "heat". Using the extended substitution axiom (α_8) can be transferred to the target domain and states that the total amount of heat stays constant over time. In a similar way, the symbol *footprint* can induce the new concept "specific heat capacity".

A.2 Analogy between the Electric and the Water Circuit.

The analogy between an electric circuit and waterflow is another classical example taken from high-school level physics. The following figure visualizes the analogy.

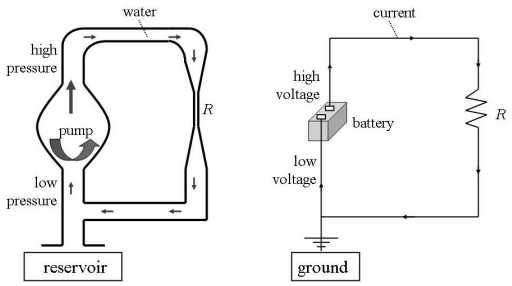
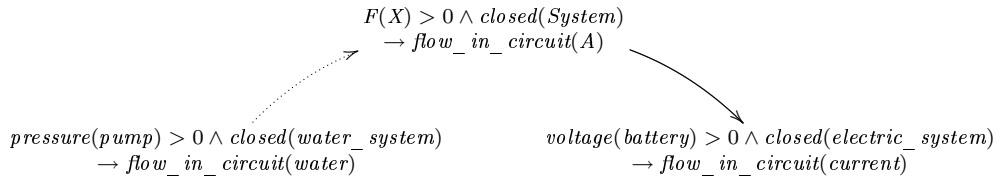


Fig. A.1. Analogy between an electric circuit and waterflow.

Waterflow System	Electric Circuit
sorts $real$, $object$, $massterm$ **entities** $water\_system : object$, $water : massterm$, $pump : object$ **functions** $presure : object \rightarrow real \times \{N/m^2\}$ **predicates** $closed : object$ $water\_circuit : object$ $flow\_in\_circuit : object$ **facts** $\alpha_1 : closed(water\_system)$ $\alpha_2 : water\_circuit(water\_system,\ water)$ $\alpha_3 : pressure(pump) > 0$ **laws** $\alpha_4 : pressure(pump) > 0 \wedge$ $closed(water\_system)$ $\rightarrow flow\_in\_circuit(water)$	**sorts** $real$, $object$, $massterm$ **entities** $electric\_system : object$, $current : massterm$, $battery : object$ **functions** $voltage : object \rightarrow real \times \{V\}$ **predicates** $closed : object$ $electric\_circuit : object$ **facts** $\beta_1 : closed(electric\_system)$ $\beta_2 : electric\_circuit(electric\_system,\ current)$ $\beta_3 : voltage(battery) > 0$

Generalization	Anti-Unifier and Substitutions
sorts $real$, $object$, $massterm$ **entities** $System : object$, $A : massterm$, $X : object$ **predicates** $closed : object$ $Circuit : object$ **facts** $\gamma_1 : closed(System)$ $\gamma_2 : Circuit(System,\ A)$ $\gamma_3 : F(X) > 0$ $\gamma_4 * : F(X) > 0 \wedge closed(System)$ $\rightarrow flow\_in\_circuit(A)$	$System \rightarrow water\_system/electric\_system$ $A \rightarrow water/current$ $X \rightarrow pump/battery$ $Circuit \rightarrow water\_circuit/electric\_circuit$ $F \rightarrow pressure/voltage$

The remaining axioms from the source domain can be transferred to the target domain as follows:



A.3 Political Analogy

The following analogy compares the structure of the Germany government with the American government. Knowledge about the source and about the target domain are equivalently available. Therefore, we give only the general-

ization specifying the analogical alignment, but there is no transfer.

German Government	US Parliament
sort *object* **entities** *bundestag*, *bundesrat*, *chancellor*, *court*, *fischer*, *government*, *minister*, *people*, *president*, *schroeder* **facts** *elect*(*people*, *bundestag*) *make_laws*(*bundestag*) *ratify_laws*(*bundesrat*) *execute_laws*(*courts*) *sign_laws*(*president*) *elect*(*bundestag*, *chancellor*) *nominates*(*chancellor*, *minister*) *is*(*chancellor*, *schroeder*) *is*(*minister*, *fischer*) *controls*(*bundestag*, *government*) *member*(*minister*, *government*) *chief*(*chancellor*, *government*)	**sort** *object* **entities** *bush*, *congress*, *court*, *house*, *judge*, *minister*, *people*, *powell*, *president*, *senate* **facts** *elect*(*people*, *congress*) *make_laws*(*house*) *ratify_laws*(*senate*) *execute_laws*(*courts*) *sign_laws*(*president*) *elect*(*people*, *president*) *nominates*(*president*, *minister*) *is*(*president*, *bush*) *is*(*minister*, *powell*) *controls*(*congress*, *president*) *member*(*minister*, *government*) *chief*(*president*, *government*)

Generalization	Generalization with Substitutions
sort *object* **entities** *people*, *congress*, *house*, *senate*, *courts*, *president*, *minister* **facts** *elect*(*people*, *A*) *make_laws*(*B*) *ratify_laws*(*C*) *execute_laws*(*courts*) *sign_laws*(*president*) *elect*(*D*, *E*) *nominates*(*E*, *minister*) *is*(*E*, *F*) *is*(*minister*, *G*) *controls*(*A*, *H*) *member*(*minister*, *government*) *chief*(*E*, *government*)	$A \rightarrow$ *bundestag/congress* $B \rightarrow$ *bundestag/house* $C \rightarrow$ *bundesrat/senate* $D \rightarrow$ *people/parliament* $E \rightarrow$ *chancelor/president* $F \rightarrow$ *schroeder/bush* $G \rightarrow$ *fischer/powell* $H \rightarrow$ *government/president* The following entities are directly mapped on each other: *government/government* *minister/minister* *people/people* *president/president*

In this analogy, HDTP discovered one-to-one mappings for the following anti-unifiers: B, C, F and G. But there are also several many-to-many mappings, for example while American *people* map to German *people* since they both elect the *bundestag/congress*, American *people* directly elect the president. In Germany, the *chancelor* is elected by the *bundestag*. All mappings can be seen the the figure above.

References

[1] D. J. Chalmers, R. M. French, D. R. Hofstadter, High-level perception, representation, and analogy: A critique of artificial intelligence methodology,

Journal of Experimental and Theoretical Artificial Intelligence 4 (3) (1992) 185–211.

[2] J. Colhoun, D. Gentner, J. Loewenstein, Learning abstract principles through principle-case comparison, in: B. Love, K. McRae, V. M. Sloutski (eds.), 30th Annual Conference of the Cognitive Science Society, Cognitive Science Society, Washington, 2008.

[3] M. Csikszentmihalyi, Creativity, in: R. A. Wilson, F. C. Keil (eds.), The MIT Encyclopaedia of the Cognitive Sciences, MIT Press, 1999.

[4] M. Dastani, B. Indurkhya, An algebraic approach to similarity and categorization (SIMCAT'97), in: M. Ramscar, U. Hahn, E. Cambouropolos, H. Pain (eds.), Interdisciplinary Workshop On Similarity and Categorization, Edinburgh, Schotland, 1997.

[5] B. Falkenhainer, Analogical interpretation in context, in: 12th Annual Conference of the Cogntive Science Society, Lawrence Erlbaum Associates, Cambridte, MA, 1990.

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

[7] R. W. Ferguson, On the mediation of levels of similarity in analogical comparisons, in: A. Schwering, U. Krumnack, K.-U. Kühnberger, H. Gust (eds.), Analogies: Integrating Multiple Cognitive Abilities (AnICA07), Publications of the Institute of Cognitive Science, University of Osnabrueck, Nashville, TN, USA, 2007.

[8] K. D. Forbus, C. Riesbeck, L. Birnbaum, K. Livingston, A. Sharman, L. Ureel, Integrating natural language, knowledge representation and reasoning, and analogical processing to learn by reading, in: Proceedings of AAAI-07: Twenty-Second Conference on Artificial Intelligence, 2007.

[9] R. Gayler, Vector symbolic architectures answer Jackendoff's challenges for cognitive neuroscience, in: P. Slezak (ed.), International Conference on Cognitive Science (ICCS/ASCS), Sydney, Australia, 2003.

[10] D. Gentner, Structure-mapping: A theoretical framework for analogy, Cognitive Science 7 (2) (1983) 155–170.

[11] D. Gentner, The mechanism of analogical learning, in: S. Vosniadou, A. Ortony (eds.), Similarity and analogical reasoning, Cambridge University Press, New York, USA, 1989, pp. 199–241.

[12] D. Gentner, K. D. Forbus, MAC/FAC: A model of similarity-based retrieval, in: 13th Annual Conference of the Cognitive Science Society, Erlbaum, Hilldale, Chicago, 1991.

[13] D. Gentner, K. J. Kurtz, Relational categories, in: W.-K. Ahn, R. L. Goldstone, B. C. Love, A. B. Markman, P. W. Wolff (eds.), Categorization inside and outside the laboratory, APA, Wasington DC, 2005, pp. 151–175.

[14] H. Gust, U. Krumnack, K.-U. Kühnberger, A. Schwering, An approach to the semantics of analogical relations, in: 2nd European Cognitive Science Conference, Lawrence Erlbaum, Delphi, Greece, 2007.

[15] H. Gust, U. Krumnack, K.-U. Kühnberger, A. Schwering, Integrating analogical and inductive learning at different levels of generalization, in: Workshop on Learning from Non-Vectorial Data (LNVD2007) at the 30th Annual German Conference on Artificial Intelligence (KI07), Publications of the Institute of Cognitive Science (PICS), University of Osnabrück, Osnabrück, 2007.

[16] H. Gust, U. Krumnack, K.-U. Kühnberger, A. Schwering, Analogical reasoning: A core of cognition, Zeitschrift für Künstliche Intelligenz (KI), Themenheft KI und Kognition.

[17] H. Gust, K.-U. Kühnberger, U. Schmid, Metaphors and heuristic-driven theory projection (HDTP), Theoretical Computer Science 354 (1) (2006) 98–117.

[18] T. Hinrichs, K. Forbus, Analogical learning in a turn-based strategy game, in: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence, 2007.

[19] D. R. Hofstadter, J. Mitchell, The copycat project: A model of mental fluidity and analogy-making, in: D. R. Hofstadter, Fluid Analogies Research Group (eds.), Fluid Concepts and Creative Analogies, Basic Books, 1995, pp. 205–267.

[20] Hofstadter, D. R. and Fluid Analogies Research Group, Fluid concepts and creative analogies, New York, 1995.

[21] K. Holyoak, P. Thagard, Analogical mapping by constraint satisfaction, Cognitive Science 13 (3) (1989) 295–335.

[22] J. E. Hummel, K. Holyoak, LISA: A computational model of analogical inference and schema induction, in: Proceedings of 16th Meeting of the Cognitive Science Society, 1996.

[23] J. E. Hummel, K. Holyoak, Distributed representations of structure: A theory of analogical access and mapping, Psychological Review 104 (3) (1997) 427–466.

[24] J. E. Hummel, K. Holyoak, A symbolic-connectionist theory of relational inference and generalization, Psychological Review 110 (2) (2003) 220–264.

[25] B. Indurkhya, Metaphor and cognition, Kluver, Dodrecht, 1992.

[26] B. Indurkhya, A computational perspective on similarity-creating metaphor, in: S. Hockey, N. Ide (eds.), Research in Humanities Computing, vol. 3, Clarendon Press, Oxford, UK, 1994, pp. 145–162.

[27] B. Indurkhya, Rationality and reasoning with metaphors, New Ideas in Psychology 25 (1) (2007) 16–36.

[28] P. Johnson-Laird, Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness, Cambridge, Mass., 1983.

[29] M. Klenk, K. D. Forbus, Cross domain analogies for learning domain theories, in: A. Schwering, U. Krumnack, K.-U. Kühnberger, H. Gust (eds.), Analogies: Integrating Multiple Cognitive Abilities (AnICA07), vol. 5-2007 of Publications of the Institute of Cognitive Science (PICS), University of Osnabrück, 2007.

[30] B. Kokinov, The dual cognitive architecture: A hybrid multi-agent approach, in: A. Cohn (ed.), 11th European Conference on Artificial Intelligence (ECAI), John Wiley & Sons, Amsterdam, 1994.

[31] U. Krumnack, A. Schwering, H. Gust, K.-U. Kühnberger, Restricted higher-order anti-unification for analogy making, in: 20th Australian Joint Conference on Artificial Intelligence (AI07), vol. 4830 of Lecture Notes of Artificial Intelligence, Springer, Gold Coast, Australia, 2007.

[32] S. E. Kuehne, K. D. Forbus, D. Gentner, B. Quinn, Seql: Category learning as progressive abstraction using structure mapping, in: Cognitive Science (CogSci2000), Philadelphia, PA, 2000.

[33] K.-U. Kühnberger, T. Wandmacher, A. Schwering, E. Ovchinnikova, U. Krumnack, H. Gust, P. Geibel, I-Cog: A computational framework for integrated cognition of higher cognitive abilities, in: A. Gelbukh, M. Morales, A. Fernando (eds.), 6th Mexican International Conference on Artificial Intelligence (MICAI), vol. 4827 of Lecture Notes of Artificial Intelligence, Springer, Aguascalientes, Mexiko, 2007.

[34] K. Lorenz, Analogy as a source of knowledge, Science 185 (147) (1974) 229–234.

[35] S. O'Hara, A model of the "redescription" process in the context of geometric proportional analogy problems, in: K. P. Jantke (ed.), International Workshop on Analogical and Inductive Inference, vol. 642 of Lecture Notes of Computer Science, Springer, 1992.

[36] G. Petkov, T. Naydenov, M. Grinberg, B. Kokinov, Building robots with analogy-based anticipation, in: KI 2006: Advances in Artificial Intelligence, vol. 4314, Springer, Bremen, Germany, 2006.

[37] G. Petkov, L. Shahbazyan, The recmap model of active recognition based on analogical mapping, in: 8th International Conference on Cognitive Modeling., Taylor & Francis, Ann Arbor, Michigan, 2007.

[38] G. D. Plotkin, A note on inductive generalization, Machine Intelligence 5 (1970) 153–163.

[39] E. Rutherford, The scattering of α and β particles by matter and the structure of the atom, Philosophical Magazine Series 6 21 669–688.

[40] A. Schwering, U. Krumnack, K.-U. Kühnberger, H. Gust, Analogical reasoning with SMT and HDTP, in: 2nd European Cognitive Science Conference (EuroCogSci07), Lawrence Erlbaum, Delphi, Greece, 2007.

[41] P. Thagard, K. Holyoak, G. Nelson, D. Gochfeld, Analogue retrieval by constraint satisfaction, Artificial Intelligence 46 (1990) 259–310.

[42] J. Yan, K. D. Forbus, D. Gentner, A theory of rerepresentation in analogical matching, in: R. Alterman, D. Kirsh (eds.), of the Twenty-fifth Annual Meeting of the Cognitive Science Society, Cognitive Science Society, Boston, Massachusetts, USA, 2003.