A Logical Approach to Reasoning by Analogy

Todd R. Davies
Artificial Intelligence Center
SRI International
Menlo Park, CA 94025, USA

Stuart J. Russell
Computer Science Division
University of California
Berkeley, CA 94720, USA

Abstract

We analyze the logical form of the domain knowledge that grounds analogical inferences and generalizations from a single instance. The form of the assumptions which justify analogies is given schematically as the "determination rule", so called because it expresses the relation of one set of variables determining the values of another set. The determination relation is a logical generalization of the different types of dependency relations defined in database theory. Specifically, we define determination as a relation between schemata of first order logic that have two kinds of free variables: (1) object variables and (2) what we call "polar" variables, which hold the place of truth values. Determination rules facilitate sound rule inference and valid conclusions projected by analogy from single instances, without implying what the conclusion should be prior to an inspection of the instance. They also provide a way to specify what information is sufficiently relevant to decide a question, prior to knowledge of the answer to the question.[1]

1 Introduction to the Problem

In this paper we consider the conditions under which propositions inferred by analogy are true or sound. As such, we are concerned with normative criteria for analogical transfer rather than a descriptive or heuristic theory. The goal is to provide a reliable, programmable strategy that will enable a system to draw conclusions by analogy only when it should.

Reasoning by analogy may be defined as the process of inferring that a *conclusion property* Q holds of a particular situation or object T (the *target*) from the fact that T shares a property or set of properties P with another situation/object S (the *source*) that has property Q. The set of common properties P is the *similarity* between S and T, and the conclusion property Q is *projected* from S onto T. The process may be summarized schematically as follows:

$$\begin{array}{l} P(S) \wedge Q(S) \\ \underline{P(T) \qquad\quad} \\ Q(T). \end{array}$$

This form of argument is nondeductive, in that its conclusion does not follow syntactically just from its premises. Instances of this argument form vary greatly in cogency. Bob's car and John's car share the property of being 1982 Mustang GLX V6 hatchbacks, but we could not infer that Bob's car is painted red just because John's car is painted red. The fact that John's car is worth about $3500 is, however, a good indication that Bob's car is worth about $3500. In the former example, the inference is not compelling; in the latter it is very probable, but the premises are true in both examples. Clearly the plausibility of the conclusion depends on information that is not provided in the premises. So the justification aspect of the logical problem of analogy, which has been much studied in the field of philosophy (see, e.g. [5], [13], [16], [31]), may be defined as follows:

THE JUSTIFICATION PROBLEM:

> Find a criterion which, if satisfied by any particular analogical inference, sufficiently establishes the truth of that inference.

Specifically, we take this to be the task of specifying background knowledge that, when added to the premises of the analogy, makes the conclusion follow soundly.

It might be noticed that the analogy process defined above can be broken down into a two-step argument as follows: (1) From the first premise $P(S) \wedge Q(S)$, conclude the *generalization* $\forall x\, P(x) \Rightarrow Q(x)$, and (2) instantiate the generalization to T and apply modus ponens to get the conclusion $Q(T)$. In this process, only the first step is nondeductive, so it looks as if the problem of justifying the analogy has been reduced to the problem of justifying a single-instance inductive generalization. The traditional criteria for evaluating the cogency of enumerative induction, however, tell us only that the inference increases in plausibility as the number of instances confirming the generalization increases (without counter-examples) and is dependent on the conclusion property being "projectible" (see [11]). If this is the only criterion applied to analogical inferences, then all projectible conclusions by analogy without counter-examples should be equally plausible, which is not the case. For example, if inspection of a red

[1] This research was made possible in part by a grant from the System Development Foundation to the Center for the Study of Language and Information, and in part by the Office of Naval Research under Contracts N00014-85-C-0013 and N00014-81-K-0004. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Office of Naval Research or the United States Government. This research was done while the second author was a student in the Computer Science Department at Stanford University, supported by a NATO studentship from the UK Science and Engineering Research Council. The first author is presently also affiliated with the Psychology Department at Stanford University.

robin reveals that its legs are longer than its beak, a projection of this conclusion onto unseen red robins is plausible, but projecting that the scratch on the first bird's beak will be observed on a second red robin is implausible. A person who has looked closely at the beak of only one red robin will have no counter-examples to either conclusion, and both conclusion properties are projectible, so the difference in cogency must be accounted for by some other criterion. The problem of analogy is thus distinct from the problem of enumerative induction because the former requires a stronger criterion for plausibility.

One approach to the analogy problem has been to regard the conclusion as plausible in proportion to the *amount* of similarity that exists between the target and the source (see [19]). Heuristic variants of this have been popular in research on analogy in AI (see, e.g. [32] and [3]). Such similarity-based methods, although intuitively appealing, suffer from some serious drawbacks. Consider again the problem of inferring properties of an unseen red robin from those of one already studied: the amount of similarity is fixed, namely that both things are red robins, but we are much happier to infer that the bodily proportions will be the same in both cases than to infer that the unseen robin will also have a scratched beak. In other words, the amount of similarity is clearly an insufficient guide to the plausibility of an analogical inference. Recognizing this, researchers studying analogy have adverted to *relevance* as an important condition on the relation between the similarity and the conclusion ([27], [15]).

To be a useful criterion, the condition of the similarity P being relevant to the conclusion Q needs to be weaker than the rule $\forall x\, P(x) \Rightarrow Q(x)$, for otherwise the conclusion in plausible analogies would always follow just by application of the rule to the target. Inspection of the source would then be redundant. So a solution to the logical problem of analogy must, in addition to providing a justification for the conclusion, also ensure that the information provided by the source instance is used in the inference. We therefore have the following:

THE NON-REDUNDANCY PROBLEM:

> The background knowledge that justifies an analogy or single-instance generalization should be insufficient to imply the conclusion given information only about the target. The source instance should provide information not otherwise contained in the database.

This condition rules out trivial solutions to the justification problem. In particular, though the additional premise $\forall x\, P(x) \Rightarrow Q(x)$ is sufficient for the truth of the inference, it does not solve the non-redundancy problem and is therefore inadequate as a general solution to the logical problem of analogy. To return to the example of Bob's and John's cars, the non-redundancy requirement stipulates that it should not be possible, merely from knowing that John's car is a 1982 Mustang GLX V6 hatchback and some rules for calculating current value, to conclude that the value of John's car is about \$3500–for then it would be unnecessary to invoke the information that Bob's car is worth that amount. The role of the source analogue (or instance) would in that case be just to point to a conclusion which could then be verified independently by applying general knowledge directly to John's car. The non-redundancy requirement assumes, by contrast, that the information provided by the source instance is not implicit in other knowledge. This requirement is important if reasoning from instances is to provide us with any conclusions that could not be inferred otherwise.

This seems like an opportune place to draw a distinction between this work and that of many others researching analogy. There has been a good deal of fruitful work on different methods for learning by analogy ([12], [2], [3], [32], [1], [15], [10]) in which the logical problem is of secondary importance to the empirical usefulness of the methods for particular domains. Similarity measures, for instance, can prove to be a successful guide to analogizing when precise relevance information is unavailable ([24]). However, when studying any form of inference, it behooves the researcher to at least consider what the basis of the inference process might be; for the most part such consideration has been lacking, with the result that analogy systems have yet to demonstrate any wide applicability or reliable performance. Our project is to provide an underlying justification for the plausibility of analogy from a logical perspective, and in so doing to provide a way to specify background knowledge that is sufficient for drawing reliable analogical inferences. The approach is intended to complement, rather than to compete with, more heuristic methods.

2 Determination Rules as a Solution

If we think about the example of the two cars (Bob's and John's), it seems clear that, while we may not know what the value of a 1986 Mustang GLX V6 hatchback is prior to knowing the value of Bob's car, we do know that the fact that a car is a Mustang GLX V6 hatchback is sufficient to *determine* its value. Abstractly, we know that either all objects with property P also have property Q, or that none do:

$$(*) \qquad (\forall x\, P(x) \Rightarrow Q(x)) \vee (\forall x P(x) \Rightarrow \neg Q(x)).$$

Having this assumption in a background theory is sufficient to guarantee the truth of the conclusion $Q(T)$ from $P(S) \wedge P(T) \wedge Q(S)$ while at the same time requiring an inspection of the source S to rule out one of the disjuncts. It is therefore a solution to both the justification problem and the non-redundancy problem.

As a way of describing the relation between P and Q in the above disjunction, we might say that P *decides* whether Q is true for any situation x. Of course, one might notice that the background knowledge we bring to the car example is more general in form. Specifically, we have knowledge of what is called in database theory a "dependency" relation ([28]), that the make, model, design, engine, condition, and year of a car determine its current value. Abstractly, a functional dependency is defined as follows ([29]):

$$(**) \qquad \forall x, y\, F(x) = F(y) \Rightarrow G(x) = G(y).$$

In this case, we say that a function (or set of functions) F *functionally determines* the value of function(s) G because the value assignment for F is associated with a unique value assignment for G. We may know this to be true without knowing exactly which value for G goes with a particular value for F. A taxonomy of the forms for the relation "$F(x)$ determines $G(x)$" has been worked out by researchers in database theory, in which such dependencies are used as integrity constraints ([28]). If the example of Bob's and John's cars (Car_B and Car_J respectively) from above is written in functional terms, as follows:

$$Make(Car_B) = Ford \wedge Make(Car_J) = Ford$$
$$Model(Car_B) = Mustang \wedge Model(Car_J) = Mustang$$
$$Design(Car_B) = GLX \wedge Design(Car_J) = GLX$$
$$Engine(Car_B) = V6 \wedge Engine(Car_J) = V6$$
$$Condition(Car_B) = Good \wedge Condition(Car_J) = Good$$
$$Year(Car_B) = 1982 \wedge Year(Car_J) = 1982$$
$$\underline{Value(Car_B) = \$3500}$$
$$Value(Car_J) = \$3500,$$

then knowing that the make, model, design, engine, condition, and year determine value thus makes the conclusion valid. In our generalized logical definition of determination (see the section on "Representation and Semantics"), the forms (*) and (**) are subsumed as special cases of a single relation "P *determines* Q", written as $P \succ Q$.

Assertions of the form "P determines Q" are actually quite common in ordinary language. When we say "The IRS decides whether you get a tax refund", or "What school you attend determines what courses are available", or, quoting a recent television advertisement, "It's when you start to save that decides where in the world you can retire to", we are expressing an invariant relation more complicated than a purely implicational rule. At the same time, we are expressing weaker information than is contained in the statement that P implies Q. If P implies Q then P determines Q, but the reverse is not true, so traditional implication falls out as a special case of determination. That the knowledge of a determination rule is what underlies preferred analogical inferences seems relatively transparent once the problem is set up as we have done. We therefore find it surprising that only recently has the possibility of valid reasoning by analogy been recognized (in [30]) and the logical form of its justification been worked out in a way that solves the non-redundancy propblem (in [6]). Most research on analogy and generalization seems to have assumed that an instance can provide at most inductive support for a rule. Our work suggests that rule formation and analogical projection are better viewed as being guided by higher level domain knowledge about what *sorts* of generalizations can be inferred from an instance. This perspective seems consistent with more recent AI techniques for doing induction and analogy (e.g. [14],[15]) which view such inferences as requiring specific knowledge about relevance rather than just an ability to evaluate similarity. We have concentrated on making the relevance criterion deductive.

3 Representation and Semantics

To define the general logical form for determination in predicate logic, we need a representation that covers (1) determination of the truth value or polarity of an expression, as in example cases of the form "$P(x)$ *decides* whether or not $Q(x)$" (formula (*) from previous section), (2) functional determination rules like (**) above, and (3) other cases in which one expression in first order logic determines another. Rules of the first form require us to extend the notion of a first order predicate schema in the following way. Because the truth value of a first order formula cannot be a defined function within the language, we introduce the concept of a *polar variable*, which can be placed at the beginning of an expression to denote that its truth value is not being specified by the expression. For example, the notation "$i\,P(x)$" can be read "whether or not $P(x)$", and it can appear on either side of the determination relation sign "$\succ$" in a determination rule, as in

$$P_1(x) \wedge i_1 P_2(x) \succ i_2 Q(x).$$

This would be read, "$P_1(x)$ and whether or not $P_2(x)$ together jointly determine whether or not $Q(x)$," where i_1 and i_2 are polar variables.

The determination relation cannot be formulated as a connective, i.e., a relation between propositions or closed formulas. Instead, it should be thought of as a relation between predicate *schemata*, or open formulas with polar variables. For a first order language $\mathbf{L}$, the set of predicate schemata for the language may be characterized as follows. If S is a sentence (closed formula or wff) of $\mathbf{L}$, then the following operations may be applied, in order, to S to generate a predicate schema:

1. Polar variables may be placed in front of any wffs that are contained as strings in S,
2. Any object variables in S may be unbound (made free) by removing quantification for any part of S, and
3. Any object constants in S may be replaced by object variables.

All of and only the expressions generated by these rules are schemata of $\mathbf{L}$.

To motivate the definition of determination, let us turn to some example pairs of schemata for which the determination relation holds. As an example of the use of polar variables, consider the rule that, being a student athlete, one's school, year, sport, and whether one is female determine who one's coach is and whether or not one has to do sit-ups. This can be represented as follows:

EXAMPLE 1:

$$(Athlete(x) \wedge Student(x) \wedge School(x) = s \wedge Year(x) = y \wedge Sport(x) = z \wedge i_1 Female(x)) \succ (Coach(x) = c \wedge i_2 Sit-ups(x)).$$

As a second example, to illustrate that the component schemata may contain quantified variables, consider the rule that, not having any deductions, having all your income from a corporate employer, and one's income determine one's tax rate:

EXAMPLE 2:

$$(Taxpayer(x) \wedge Citizen(x, US) \wedge (\neg \exists d\, Deductions(x, d)) \wedge (\forall i\, Income(i, x) \Rightarrow Corporate(i)) \wedge PersonalIncome(x) = p) \succ (TaxRate(x) = r).$$

In each of the above examples, the free variables in the component schemata may be divided, relative to the determination rule, into a *case* set $\underline{x}$ of those that appear free in both the *determinant* (left-hand side) and the *resultant* (right-hand side), a *predictor* set $\underline{y}$ of those that appear only in the determinant schema, and a *response* set $\underline{z}$ of those that appear only in the resultant.[2]. These sets are uniquely defined for each determination rule. In particular, for example 1 they are $\underline{x} = \{x\}$, $\underline{y} = \{s, y, z, i_1\}$, and $\underline{z} = \{c, i_2\}$; and for example 2 they are $\underline{x} = \{x\}$, $\underline{y} = \{p\}$, and $\underline{z} = \{r\}$. In general, for a predicate schema Σ with free variables $\underline{x}$ and $\underline{y}$, and a predicate schema X with free variables $\underline{x}$ (shared with Σ) and $\underline{z}$ (unshared), whether the determination relation holds is defined as follows:

THE DEFINITION OF DETERMINATION:

$$\Sigma[\underline{x}, \underline{y}] \succ X[\underline{x}, \underline{z}]$$
$$\text{iff}$$
$$\forall \underline{y}, \underline{z} (\exists \underline{x}\, \Sigma[\underline{x}, \underline{y}] \wedge X[\underline{x}, \underline{z}]) \Rightarrow (\forall \underline{x}\, \Sigma[\underline{x}, \underline{y}] \Rightarrow X[\underline{x}, \underline{z}]).$$

In interpreting this formula, quantified polar variables range over the unary Boolean operators (negation and affirmation) as their domain of constants, and the standard Tarskian semantics is applied in evaluating truth in the usual way (see [9]). This definition covers the full range of determination rules expressible in first order logic, and is therefore more expressive than the set of rules restricted to dependencies between frame slots, given a fixed vocabulary of constants. Nonetheless, one way to view a predicate schema is as a frame, with slots corresponding to the free variables.

4 Use in Reasoning

Much of the work in machine learning, from the early days when Shakey was learning macro-operators for action ([21]) to more recent work on chunking ([22]) and explanation-based generalization ([20]), has involved getting systems to learn and represent explicitly rules and relations between concepts that could have been derived from the start. In Shakey's case, for example, the planning algorithm and knowledge about operators in STRIPS were a sufficient apparatus for deriving a plan to achieve a given goal. To say that Shakey "learned" a specific sequence of actions for achieving the goal means only that the plan was not derived until the goal first arose. Likewise, in EBG, explaining why the training example is an instance of a concept requires knowing beforehand that the instance embodies a set of conditions sufficient for the concept to apply, and chunking, despite its power to simplify knowledge at the appropriate level, does not in the logician's terms add knowledge to the system. By defining determination rules prior to the acquisition of case data, we can enable the system to generalize appropriately without making the rules it will generate implicit from the start.

Determination rules are the kind of knowledge that programmers of an intelligent system often have. We may not know very many specific rules about which coaches instruct which teams, but we still know that the latter determines the former, and this knowledge has the potential to generate an infinite number of more fine-grained rules. In addition to enhancing the power of intelligent systems, the logical formulation of analogical inference enables it to be used reliably in the logic programming and expert system contexts. A logic programming implementation is described in the next section. Determination rules may be useful in knowledge engineering for two reasons:

1. In many domains a strong (implicational) theory may not be available, whereas determination rules can be provided, and the system can gain expertise through the acquisition of examples from which it can reason by analogy.

2. Even when a strong theory is available, its complete elucidation may be difficult, and it may be easier to elicit knowledge using questions of the form "What are the factors which go into making decisions about Q?", i.e., to extract determination rules.

The use of determination rules appears to be a natural stage in the process of knowledge acquisition, occurring prior to the acquisition of a strong predictive theory; for example, we have as yet no theory that can even come close to predicting the vocabulary, grammar and usage of an entire language simply from facts about the nation it belongs to, but we still have the corresponding determination rule that one's nationality determines one's native language, with a few exceptions. We have been building a list of different categories of determinative knowledge. Here are some examples of processes in which determination rules are found:

- Physical processes: initial conditions determine outcome; boundary conditions determine steady-state values for whole system; biological ancestry determines gross physical structure; developmental environment determines fine structure of behavior; structure determines function; function determines structure (less strongly); disease determines symptoms; symptoms determine disease (less well); diet, exercise and genes determine weight; etc.

[2]Readers familiar with statistical modeling might notice that the terms for these sets of variables are borrowed from regression analysis. For a discussion of the statistical analogue of determination, and its relations to regression and classificiation, see [7]

- Processes performed by "rational agents": case description determines legal outcome; upbringing and education determine political leaning; social class and location determine buying patterns; nationality determines language; zip code determines state; address determines newspaper delivery time; etc.

- Processes in formal systems: program input determines program output; program specification determines program; etc.

- The system's own problem-solving processes: all the problem solving abilities the system has, be they planning, search, inference, programming or whatever, can be analyzed into an input P and an output Q. Constructive processes, such as planning and design, which have enormous search spaces, are particularly amenable to reasoning by analogy. ([4] begins to address these issues, implicitly using the determination rule that (exact) problem specification determines solution; the key issue to be resolved before such work can succeed is to identify the various abstracted levels of description for problems and solutions which will allow use of less specific determination rules that do not require exact matching of specifications.)

5 Implementation in a Logic Programming System

Determination-based analogical reasoning can be implemented directly as an extension to a logic programming system, such as Genesereth's MRS system (see [23]). The programmer simply adds whatever determination rules are available to the database and the system will use them whenever possible to perform analogical reasoning.

Given a query $X[T, \underline{z}]$, the basic procedure for solving it by analogy is as follows:

1. Find Σ such that $\Sigma[\underline{x}, \underline{y}] \succ X[\underline{x}, \underline{z}]$ (i.e., decide which facts could be relevant).

2. Find $\underline{y}$ such that $\Sigma[T, \underline{y}]$ (i.e., see how those facts are instantiated in the target).

3. Find S such that $\Sigma[S, \underline{y}]$ and $S \neq T$ (i.e., find a suitable source).

4. Find $\underline{z}$ such that $X[S, \underline{z}]$ (i.e., find the answer to the query from the source).

5. Return $\underline{z}$ as the solution to the query $X[T, \underline{z}]$.

We add this procedure to the system's recursive routine for solving a goal, so that it now has three alternatives:

1. Look up the answer in the database.

2. Backchain on an applicable implication rule.

3. Analogize using an applicable determination rule.

To solve goal $X[T, \underline{z}]$ using determination rule $\Sigma[\underline{x}, \underline{y}] \succ X[\underline{x}, \underline{z}]$, we simply add the following conjunctive goal to the agenda:

$$\Sigma[\underline{t}, \underline{y}] \wedge \Sigma[\underline{s}, \underline{y}] \wedge (\underline{s} \neq \underline{t}) \wedge X[\underline{s}, \underline{z}].$$

The subgoals of this can be solved recursively by the same three alternative methods, thus achieving the procedure given above.

An example may be helpful here. Suppose we have the goal of finding out what language Jack speaks, i.e., $NativeLanguage(Jack, z)$. We have the following background information:

$$\begin{array}{l}
Nationality(Jack, UK)\\
Male(Jack)\\
Height(Jack, 6')\\
\ldots\\
Nationality(Giuseppe, Italy)\\
Male(Giuseppe)\\
Height(Giuseppe, 6')\\
NativeLanguage(Giuseppe, Italian)\\
\ldots\\
Nationality(Jill, UK)\\
Female(Jill)\\
Height(Jill, 5'10'')\\
NativeLanguage(Jill, English)\\
\ldots
\end{array}$$

and among our determination rules we have that nationality determines native language (except for Swiss), as well as other such rules, for instance that nationality and whether or not one has dual citizenship determines whether or not one needs a visa to enter the United States and how long one may stay:

$$\begin{array}{l}
(Nationality(x, n) \wedge \neg Nationality(x, Swiss))\\
\quad \succ (NativeLanguage(x, l).\\
(Nationality(x, n) \wedge i_1 Dualcitizen(x, US))\\
\quad \succ (i_2 NeedVisa(x, US) \wedge Maxstay(x, t)).
\end{array}$$

Using the first of these determination rules, the system generates the new goal:

$$\begin{array}{l}
(Nationality(Jack, n) \wedge\\
\quad \neg Nationality(Jack, Swiss)) \wedge\\
(Nationality(s, n) \wedge \neg Nationality(s, Swiss)) \wedge\\
s \neq Jack \wedge\\
NativeLanguage(s, z),
\end{array}$$

which is solved after a few simple deduction steps, with Jill as the source s. One may observe that the more "similar" source Giuseppe is ignored, and that the irrelevant facts about Jack and Jill are not examined. When the facts satisfying the various subgoals of the analogy are not explicitly available in the database, the system will of course attempt solutions by further reasoning, either analogical or implicational. For example, if $Nationality(Jill, UK)$ were replaced by $Birthplace(Jill, London)$, then the analogy could still succeed if a rule relating *Birthplace* and *Nationality* were available. Thus we have a natural, goal-directed *reformulation* which reveals implicit similarities in an efficient manner.

In comparison to the more traditional, heuristic approaches to analogy, the use of determination rules has significant efficiency advantages in addition to its other properties. Winston ([32]) and Greiner ([12]) point out the enormous complexity of matching the target against all possible sources in all possible ways to find out the most similar source; as we observed in the implementation example, finding the determination rule first enables us to pick out the relevant target facts and use those to index directly to an appropriate source, thus overcoming the matching problem. We also render irrelevant the problem of finding a suitable similarity metric, and transform the reformulation problem (which arises when a change of representation might reveal a previously hidden similarity) from an open-ended nightmare of forward inference into a relatively controlled, goal-directed process.

The ability of determination-based analogical reasoning to avoid unnecessary matching makes it a reasonable alternative to traditional rule-based logic systems. For some problems, analogy is more efficient than using a corresponding set of implication rules. A determination rule $P(x,y) \succ Q(x,z)$ and a set of instances replace a set of implication rules:

$$\forall x\, P(x, Y_1) \Rightarrow Q(x, Z_1)$$
$$\ldots$$
$$\forall x\, P(x, Y_n) \Rightarrow Q(x, Z_n),$$

where n can be arbitrarily large. Furthermore, since it must test the premises of every rule that could imply a goal until it finds the right one, a backward chaining system requires a lengthy search that can be avoided by using a determination rule.

A common form of reasoning that displays this behavior is taxonomic inheritance, for which we might use a rule such as

$$\forall x\, IsA(x, 73DodgeVan) \Rightarrow ValueIn87(x, \$650)$$

to conclude the current resale value of one of our cars. With 7500 models in our database, this would take us 7500/2 backchains on average. Replacing the implication rules with a determination rule $IsA(x,y) \succ ValueIn86(x,z)$ and a collection of prototypical instances (exactly analogous to the TypicalElephant frames in semantic nets) we can solve our goal in four backchaining steps.

Another example is that of diagnostic reasoning, in which the (simplified) traditional approach uses a collection of rules of the form:

$$\forall x\, HasSymptoms(x, < Symptom - list_k >)$$
$$\Rightarrow HasDisease(x, < Disease_l >).$$

These implication rules would be replaced by a determination rule $HasSymptoms(x,y) \succ HasDisease(x,z)$ and a case library.

6 Conclusion

There are a number of problems related to analogy that we have not solved. What we have is a method for generating correct generalizations and analogical inferences, given correct determination rules. At the same time, our work has created new problems: a reasonable next step is to work out how determination rules can themselves be acquired. Some early thought on the determination rule acquisition problem points to four basic methods:

1. Deduce a determination rule from other known facts (For an example, see [26]).
2. Induce a determination rule from instances (essentially calculate the empirical degree of determination of X by Σ—see [25] and [7]).
3. Induce a determination rule from a collection of specific rules.
4. Generalize from a collection of more specific determination rules.

Because we have a formal definition for determination, inductive acquisition of determination rules is conceptually straightforward, if pragmatically troublesome. Acquisition experiments on a broad knowledge base are currently under way using the CYC system ([17]). We are also building determination-based expert systems by induction from examples in the domains of market forecasting and mechanical device diagnosis from acoustic emission. The results so far seem very promising.

A full understanding of the human processes of analogical inference and generalization will surely require further investigations into how we measure similarity, how situations and rules are encoded and retrieved, and what heuristics are used in projecting conclusions when a valid argument cannot be made. But it seems that logic can tell us quite a lot about analogy, by giving us a standard for evaluating the truth of its conclusions, a general form for its justification, and a language for distinguishing it from other forms of inference. At the same time, we have found a consideration of the logical problem to be of practical benefit, for reasoning by analogy using determinative knowledge appears to give a system the ability to learn reliably new rules that would otherwise need to be programmed.

7 Acknowledgments

We would like to thank our advisors, John Perry, Mike Genesereth, and Doug Lenat, as well as Doug Edwards, Bryn Ekroot, Russ Greiner, Benjamin Grosof, David Helman, Jerry Hobbs, Dikran Karagueuzian, Kurt Konolige, Stan Rosenschein, Devika Subramanian, Dirk Ruiz, Amos Tversky, Paul Rosenbloom, and J. O. Urmson for fruitful discussions, constructive criticism and moral support.

References

[1] Burstein, M. H. A Model of Incremental Analogical Reasoning and Debugging. In *Proceedings of the National Conference on Artficial Intelligence*, 1983, pp. 45-48.

[2] Carbonell, J. G. A Computational Model of Analogical Problem Solving. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, 1981, pp. 147-152.

[3] Carbonell, J. G. Derivational Analogy and Its Role in Problem Solving. In *Proceedings of the National Conference on Artificial Intelligence*, 1983, pp. 64-69.

[4] Carbonell, J. G. Derivational Analogy. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning II*, Morgan Kaufmann, 1986.

[5] Carnap, R. *Logical Foundations of Probability*. University of Chicago Press, 1963.

[6] Davies, T. *Analogy*. Undergraduate honors thesis, Stanford University, 1985. Issued as Informal Note No. IN-CSLI-85-4, Center for the Study of Language and Information, Stanford University, 1985.

[7] Davies, T. R. A Normative Theory of Generalization and Reasoning by Analogy. To appear in Helman, David H., editor, *Analogical Reasoning: Perspectives of Artificial Intelligence, Cognitive Science, and Philosophy*, D. Reidel, Forthcoming.

[8] Gallier, J. H. *Logic for Computer Science: Foundations of Automatic Theorem Proving*. Harper and Row, 1986.

[9] Genesereth, M. R. and Nilsson, N. J. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann, In Press.

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

[11] Goodman, N. *Fact, Fiction, and Forecast*. Harvard University Press, 1983.

[12] Greiner, R. *Learning by Understanding Analogies*. Ph.D. thesis, Stanford University, 1985. Issued as Technical Report No. STAN-CS-85-1071, Department of Computer Science, Stanford University, 1985.

[13] Hesse, M. *Models and Analogies in Science*. Notre Dame University Press, 1966.

[14] Holland, J., Holyoak, K., Nisbett, R., and Thagard, P. *Induction: Processes of Inference, Learning, and Discovery*. MIT Press, 1986.

[15] Kedar-Cabelli, S. Purpose-directed Analogy. In *The Seventh Annual Conference of the Cognitive Science Society*, 1985, pp. 150-159.

[16] Leblanc, H. A Rationale for Analogical Inference. *Philosophical Studies*, 20:29-31, 1969.

[17] Lenat, D. CYC: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquisition Bottlenecks. *The AI Magazine*, 6:65-85, 1986.

[18] Marciszewski, W. *Dictionary of Logic as Applied in the Study of Language*. Martinus Nijhoff Publishers, 1981.

[19] Mill, J. S. *A System of Logic*. Harper and Brothers Publishers, 1900.

[20] Mitchell, T. M., Keller, R. M., and Kedar-Cabelli, S. T. Explanation-based Generalization: A Unifying View. *Machine Learning*, 1(1), 1986.

[21] Nilsson, N. J. *Shakey the Robot*. Technical Note 323, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1984.

[22] Rosenbloom, P. S., and Newell, A. The Chunking of Goal Hierarchies: A Generalized Model of Practice. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning II*, Morgan Kaufmann, 1986.

[23] Russell, S. J. *The Compleat Guide to MRS*. Technical Report No. STAN-CS-85-1080, Department of Computer Science, Stanford University, 1985.

[24] Russell, S. J. A Quantitative Analysis of Analogy by Similarity. In *Proceedings of the National Conference on Artificial Intelligence*, 1986, pp. 284-288.

[25] Russell, S. J. *Analogical and Inductive Reasoning*. Ph.D. thesis, Stanford University, 1986.

[26] Russell, S. J., and Grosof, B. N. A Declarative Approach to Bias in Concept Learning. In *Proceedings of the National Conference on Artificial Intelligence*, 1987.

[27] Shaw, W. H. and Ashley, L. R. Analogy and Inference. *Dialogue: Canadian Journal of Philosophy*, 22:415-432, 1983.

[28] Ullman, J. D. *Principles of Database Systems*. Computer Science Press, 1983.

[29] Vardi, M. Y. *The Implication and Finite Implication Problems for Typed Template Dependencies*. Technical Report No. STAN-CS-82-912, Department of Computer Science, Stanford University, 1982.

[30] Weitzenfeld, J. S. Valid Reasoning by Analogy. *Philosophy of Science*, 51:137-149, 1984.

[31] Wilson, P. R. On the Argument by Analogy. *Philosophy of Science*, 31:34-39, 1964.

[32] Winston, P. H. Learning and Reasoning by Analogy. *Communications of the ACM*, 23:689-703, 1980.