Machine Learning 1: 47–80, 1986
© 1986 Kluwer Academic Publishers, Boston – Manufactured in The Netherlands

Explanation-Based Generalization: A Unifying View

TOM M. MITCHELL (MITCHELL @ RED.RUTGERS.EDU)
RICHARD M. KELLER (KELLER @ RED.RUTGERS.EDU)
SMADAR T. KEDAR-CABELLI (KEDAR-CABELLI @ RED.RUTGERS.EDU)
Computer Science Department, Rutgers University, New Brunswick, NJ 08903, U.S.A.

(Received August 1, 1985)

Key words: explanation-based learning, explanation-based generalization, goal regression, constraint back-propagation, operationalization, similarity-based generalization.

Abstract. The problem of formulating general concepts from specific training examples has long been a major focus of machine learning research. While most previous research has focused on empirical methods for generalizing from a large number of training examples using no domain-specific knowledge, in the past few years new methods have been developed for applying domain-specific knowledge to formulate valid generalizations from single training examples. The characteristic common to these methods is that their ability to *generalize* from a single example follows from their ability to *explain* why the training example is a member of the concept being learned. This paper proposes a general, domain-independent mechanism, called EBG, that unifies previous approaches to explanation-based generalization. The EBG .method is illustrated in the context of several example problems, and used to contrast several existing systems for explanation-based generalization. The perspective on explanation-based generalization afforded by this general method is also used to identify open research problems in this area.

1. Introduction and motivation

The ability to generalize from examples is widely recognized as an essential capability of any learning system. Generalization involves observing a set of training examples of some general concept, identifying the essential features common to these examples, then formulating a concept definition based on these common features. The generalization process can thus be viewed as a search through a vast space of possible concept definitions, in search of a correct definition of the concept to be learned. Because this space of possible concept definitions is vast, the heart of the generalization problem lies in utilizing whatever training data, assumptions and knowledge are available to constrain this search.

Most research on the generalization problem has focused on empirical, data-intensive methods that rely on large numbers of training examples to constrain the search for the correct generalization (see Mitchell, 1982; Michalski, 1983; Dietterich, 1982 for overviews of these methods). These methods all employ some kind of *inductive bias* to guide the inductive leap that they must make in order to define a concept from only a subset of its examples (Mitchell, 1980). This bias is typically built into the generalizer by providing it with knowledge only of those example features that are presumed relevant to describing the concept to be learned. Through various algorithms it is then possible to search through the restricted space of concepts definable in terms of these allowed features, to determine concept definitions consistent with the training examples. Because these methods are based on searching for features that are common to the training examples, we shall refer to them as *similarity-based* generalization methods.[1]

In recent years, a number of researchers have proposed generalization methods that contrast sharply with these data-intensive, similarity-based methods (e.g., Borgida et al., 1985; DeJong, 1983; Kedar-Cabelli, 1985; Keller, 1983; Lebowitz, 1985; Mahadevan, 1985; Minton, 1984; Mitchell, 1983; Mitchell et al., 1985; O'Rorke, 1984; Salzberg & Atkinson, 1984; Schank, 1982; Silver, 1983; Utgoff, 1983; Winston et al., 1983). Rather than relying on many training examples and an inductive bias to constrain the search for a correct generalization, these more recent methods constrain the search by relying on knowledge of the task domain and of the concept under study. After analyzing a single training example in terms of this knowledge, these methods are able to produce a valid generalization of the example *along with a deductive justification of the generalization in terms of the system's knowledge*. More precisely, these *explanation-based* methods[2] analyze the training example by first constructing an explanation of how the example satisfies the definition of the concept under study. The features of the example identified by this explanation are then used as the basis for formulating the general concept definition. The justification for this concept definition follows from the explanation constructed for the training example.

Thus, by relying on knowledge of the domain and of the concept under study, explanation-based methods overcome the fundamental difficulty associated with inductive, similarity-based methods: their inability to justify the generalizations that they produce. The basic difference between the two classes of methods is that similarity-based methods must rely on some form of inductive bias to guide generalization, whereas explanation-based methods rely instead on their domain knowledge. While explanation-based methods provide a more reliable means of

[1] The term *similarity-based generalization* was suggested by Lebowitz (1985). We use this term to cover both methods that search for similarities among positive examples, and for differences between positive and negative examples.

[2] The term *explanation-based generalization* was first introduced by DeJong (1981) to describe his particular generalization method. The authors have previously used the term *goal-directed generalization* (Mitchell, 1983) to refer to their own explanation-based generalization method. In this paper, we use the term explanation-based generalization to refer to the entire class of methods that formulate generalizations by constructing explanations.

generalization, and are able to extract more information from individual training examples, they also require that the learner possess knowledge of the domain and of the concept under study. It seems clear that for a large number of generalization problems encountered by intelligent agents, this required knowledge is available to the learner. In this paper we present and analyze a number of such generalization problems.

The purpose of this paper is to consider in detail the capabilities and requirements of explanation-based approaches to generalization, and to introduce a single mechanism that unifies previously described approaches. In particular, we present a domain-independent method (called EBG) for utilizing domain-specific knowledge to guide generalization, and illustrate its use in a number of generalization tasks that have previously been approached using differing explanation-based methods. EBG constitutes a more general mechanism for explanation-based generalization than these previous approaches. Because it requires a larger number of explicit inputs (i.e., the training example, a domain theory, a definition of the concept under study, and a description of the form in which the learned concept must be expressed) EBG can be instantiated for a wider variety of learning tasks. Finally, EBG provides a perspective for identifying the present limitations of explanation-based generalization, and for identifying open research problems in this area.

The remainder of this paper is organized as follows. Section 2 introduces the general EBG method for explanation-based generalization, and illustrates the method with an example. Section 3 then illustrates the EBG method in the context of two additional examples: (1) learning a structural definition of a cup from a training example plus knowledge about the function of a cup (based on Winston et al.'s (1983) work), and (2) learning a search heuristic from an example search tree plus knowledge about search and the search space (based on Mitchell et al.'s (1983) work on the LEX system). Section 4 concludes with a general perspective on explanation-based generalization and a discussion of significant open research issues in this area. The appendix relates DeJong's (1981, 1983) research on explanation-based generalization and explanatory schema acquisition to the other work discussed here.

2. Explanation-based generalization: discussion and an example

The key insight behind explanation-based generalization is that it is possible to form a justified generalization of a single positive training example provided the learning system is endowed with some explanatory capabilities. In particular, the system must be able to explain to itself *why* the training example is an example of the concept under study. Thus, the generalizer is presumed to possess a definition of the concept under study as well as domain knowledge for constructing the required explanation. In this section, we define more precisely the class of generalization problems covered by explanation-based methods, define the general EBG method, and illustrate it in terms of a specific example problem.

2.1 The explanation-based generalization problem

In order to define the generalization problem considered here, we first introduce some terminology. A *concept* is defined as a predicate over some universe of instances, and thus characterizes some subset of the instances. Each *instance* in this universe is described by a collection of ground literals that represent its features and their values. A *concept definition* describes the necessary and sufficient conditions for being an example of the concept, while a *sufficient concept definition* describes sufficient conditions for being an example of the concept. An instance that satisfies the concept definition is called an *example*, or *positive example* of that concept, whereas an instance that does not satisfy the concept definition is called a *negative example* of that concept. A *generalization* of an example is a concept definition which describes a set containing that example.[3] An *explanation* of how an instance is an example of a concept is a proof that the example satisfies the concept definition. An *explanation structure* is the proof tree, modified by replacing each instantiated rule by the associated general rule.

The generic problem definition shown in Table 1 summarizes the class of generalization problems considered in this paper. Table 2 illustrates a particular instance of an explanation-based generalization problem from this class. As indicated by these tables, defining an explanation-based learning problem involves specifying four kinds of information:

- The *goal concept* defines the concept to be acquired. For instance, in the problem defined in Table 2 the task is to learn to recognize pairs of objects <x,y> such that it is safe to stack x on top of y. Notice that the goal concept

Table 1. The explanation-based generalization problem

Given:

- *Goal Concept:* A concept definition describing the concept to be learned. (It is assumed that this concept definition fails to satisfy the Operationality Criterion.)
- *Training Example:* An example of the goal concept.
- *Domain Theory:* A set of rules and facts to be used in explaining how the training example is an example of the goal concept.
- *Operationality Criterion:* A predicate over concept definitions, specifying the form in which the learned concept definition must be expressed.

Determine:

- A generalization of the training example that is a sufficient concept definition for the goal concept and that satisfies the operationality criterion.

[3] In fact, we use the term *generalization* in this paper both as a noun (to refer to a general concept definition), and as a verb (to refer to the process of deriving this generalization).

Table 2. The SAFE-TO-STACK generalization problem after Borgida et al. (1985)

Given:

- *Goal Concept:* Pairs of objects <x, y> such that SAFE-TO-STACK (x, y), where
  SAFE-TO-STACK (x, y) ⇔ NOT (FRAGILE (y)) ∨ LIGHTER (x, y).
- *Training Example:*
  ON (OBJ1, OBJ2)
  ISA (OBJ1, BOX)
  ISA (OBJ2, ENDTABLE)
  COLOR (OBJ1, RED)
  COLOR (OBJ2, BLUE)
  VOLUME (OBJ1, 1)
  DENSITY (OBJ1, .1)
  . . .
- *Domain Theory:*
  VOLUME (p1, v1) ∧ DENSITY (p1, d1) → WEIGHT (p1, v1*d1)
  WEIGHT (p1, w1) ∧ WEIGHT (p2, w2) ∧ LESS (w1, w2) → LIGHTER (p1, p2)
  ISA (p1, ENDTABLE) → WEIGHT (p1, 5) (default)
  LESS (.1, 5)
  . . .
- *Operationality Criterion:* The concept definition must be expressed in terms of the predicates used to describe examples (e.g., VOLUME, COLOR, DENSITY) or other selected, easily evaluated, predicates from the domain theory (e.g., LESS).

Determine:

- A generalization of training example that is a sufficient concept definition for the goal concept and that satisfies the operationality criterion.

here, SAFE-TO-STACK, is defined in terms of the predicates FRAGILE and LIGHTER, whereas the training example is defined in terms of other predicates (i.e., COLOR, DENSITY, VOLUME, etc.).

- The *training example* is a positive example of the goal concept. For instance, the training example of Table 2 describes a pair of objects, a box and an end-table, where one is safely stacked on the other.
- The *domain theory* includes a set of rules and facts that allow explaining how training examples are members of the goal concept. For instance, the domain theory for this problem includes definitions of FRAGILE and LIGHTER, rules for inferring features like the WEIGHT of an object from its DENSITY and VOLUME, rules that suggest default values such as the WEIGHT of an ENDTABLE, and facts such as '.1 is LESS than 5'.
- The *operationality criterion* defines the terms in which the output concept definition must be expressed. Our use of this term is based on Mostow's (1981) definition that a procedure is *operational* relative to a given agent and task, provided that the procedure can be applied by the agent to solve the task. Similarly, we assume that the learned concept definition will be used by some agent to perform some task, and must be defined in terms operational for

that agent and task. For this problem, the operationality criterion requires that the final concept definition be described in terms of the predicates used to describe the training example (e.g., COLOR, VOLUME, DENSITY) or in terms of a selected set of easily evaluated predicates from the domain theory (e.g., LESS). Reexpressing the goal concept in these terms will make it operational with respect to the task of *efficiently recognizing examples of the concept.*

Given these four inputs, the task is to determine a generalization of the *training example* that is a sufficient concept definition for the *goal concept* and that satisfies the *operationality criterion*. Note that the notion of operationality is crucial for explanation-based generalization: if the operationality criterion were not specified, the input goal concept definition could always be a correct output concept definition and there would be nothing to learn! The operationality criterion imposes a requirement that learned concept definitions must be not only correct, but also *in a usable form* before learning is complete. This additional requirement is based on the viewpoint that concept definitions are not learned as theoretical entities, but rather as practical entities to be used by a particular agent for a particular task.

2.2 The EBG method

The EBG method, which is designed to address the above class of problems, is defined as follows:

The EBG method

1. *Explain:* Construct an explanation in terms of the *domain theory* that proves how the *training example* satisfies the *goal concept* definition.

   - This explanation must be constructed so that each branch of the explanation structure terminates in an expression that satisfies the *operationality criterion.*

2. *Generalize:* Determine a set of sufficient conditions under which the explanation structure holds, stated in terms that satisfy the *operationality criterion.*

   - This is accomplished by regressing the *goal concept* through the explanation structure. The conjunction of the resulting regressed expressions constitutes the desired concept definition.

To see more concretely how the EBG method works, consider again the problem of learning the concept SAFE-TO-STACK (x, y). The bottom of Figure 1 shows a training example for this problem, described in terms of a semantic network of objects and relations. In particular, the example consists of two physical objects, OBJ1

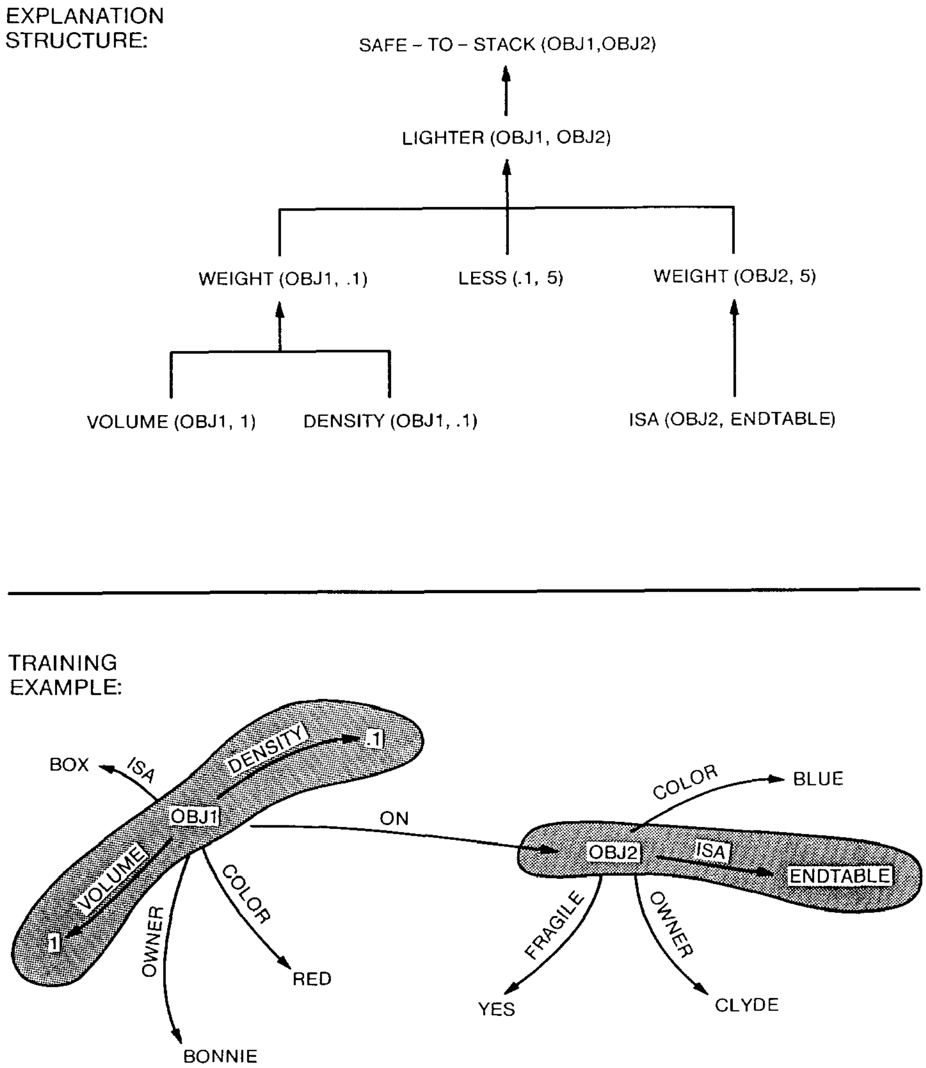

Figure 1. Explanation of SAFE-TO-STACK (OBJ1, OBJ2).

and OBJ2, in which OBJ1 is ON OBJ2, and for which several features of the objects are described (e.g., their OWNERs, COLORs).

Given this training example, the task is to determine which of its features are relevant to characterizing the goal concept, and which are irrelevant. To this end, the first step of the EBG method is to construct an explanation of how the training example satisfies the goal concept. Notice that the explanation constitutes a proof, and constructing such an explanation therefore may involve in general the complexities of theorem proving. The explanation for the training example depicted in the lower portion of Figure 1 is given in the top portion of the figure. As shown there, the pair of objects <OBJ1, OBJ2> satisfies the goal concept SAFE-TO-STACK because OBJ1 is LIGHTER than OBJ2. Furthermore, this is known because the WEIGHTs of OBJ1 and OBJ2 can be inferred. For OBJ1, the WEIGHT is inferred from its DENSITY and VOLUME, whereas for OBJ2 the WEIGHT is inferred based on a rule that specifies the default weight of ENDTABLEs in general.

Through this chain of inferences, the explanation structure demonstrates how OBJ1 and OBJ2 satisfy the goal concept definition. Note that the explanation structure has been constructed so that each of its branches terminates in an expression that satisfies the operationality criterion (e.g., VOLUME (OBJ1, 1), LESS (.1, 5)). In this way, the explanation structure singles out those features of the training example that are relevant to satisfying the goal concept, and that provide the basis for constructing a justified generalization of the training example. For the current example, these relevant training example features are shown shaded over in the figure, and correspond to the conjunction VOLUME (OBJ1, 1) $\wedge$ DENSITY (OBJ1, 0.1) $\wedge$ ISA (OBJ2, ENDTABLE).

Whereas the first step of the EBG method isolates the relevant features of the training example, it does not determine the desired generalized constraints on feature values. For instance, while the feature VOLUME (OBJ1, 1) is relevant to explaining how the present training example satisfies the goal concept, the general constraint on acceptable values for VOLUME is yet to be determined. The second step of the EBG method therefore generalizes on those feature values selected by the first step, by determining sufficient conditions on these feature values that allow each step in the explanation structure to carry through.

In order to determine general sufficient conditions under which the explanation holds, the second step of the EBG method involves regressing (back propagating) the goal concept step by step back through the explanation structure. In general, *regressing* a given formula F through a rule R is a mechanism for determining the necessary and sufficient (weakest) conditions under which that rule R can be used to infer F. We employ a slightly modified version of the goal-regression algorithm described by Waldinger (1977) and Nilsson (1980).[4] Our modified goal regression algorithm computes an expression that represents only a sufficient condition (rather than necessary and sufficient conditions) under which rule R can be used to infer formula F, but that corresponds closely to the training example under consideration. In particular, whereas the general goal regression algorithm considers all possible variable bindings (unifications) under which R can infer F, our modified algorithm considers only the specific variable bindings used in the explanation of the training example. Furthermore, if the rule R contains a disjunctive antecedent (left-hand side), then our

Dijkstra (1976) introduces the related notion of *weakest preconditions* in the context of proving program correctness. The *weakest preconditions* of a program characterize the set of all initial states of that program such that activation guarantees a final state satisfying some postcondition.

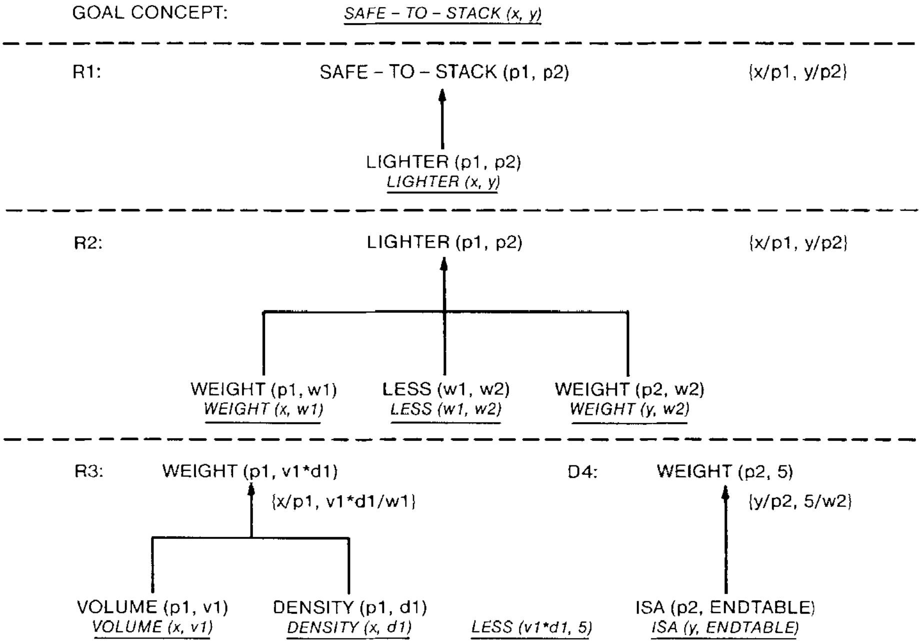

Figure 2. Generalizing from the explanation of SAFE-TO-STACK (OBJ1, OBJ2). (Underlined expressions are the results of regressing the goal concept.)

algorithm considers only the particular disjuncts satisfied by the training example.

Figure 2 illustrates the second step of the EBG method in the context of the SAFE-TO-STACK example. In the first (topmost) step of this figure, the goal concept expression SAFE-TO-STACK (x, y) is regressed through the rule LIGHTER (p1, p2) → SAFE-TO-STACK (p1, p2),[5] to determine that LIGHTER (x, y) is a sufficient condition for inferring SAFE-TO-STACK (x, y). Similarly, regressing LIGHTER (x, y) through the next step in the explanation structure yields the expression WEIGHT (x, w1) ∧ WEIGHT (y, w2) ∧ LESS (w1, w2). This expression is in turn regressed through the final steps of the explanation structure to yield the operational definition for SAFE-TO-STACK (x, y).

To illustrate the goal regression process in greater detail, consider the final step of Figure 2 in which the expression WEIGHT (x, w1) ∧ WEIGHT (y, w2) ∧ LESS (w1, w2) is regressed through the final steps of the explanation structure. Each conjunct of the expression is regressed separately through the appropriate rule, in the following way. The conjunct is unified (matched) with the consequent (right-hand side) of

[5] Notice that the definition of SAFE-TO-STACK given in Table 2 is a disjunctive definition. As noted above, the procedure considers only the disjunct that is satisfied by the current training example (e.g., the disjunct involving the LIGHTER predicate).

the rule to yield some set of substitutions (particular variable bindings). The substitution consistent with the example is then applied to the antecedent (left-hand side) of the rule to yield the resulting regressed expression.[6] Any conjuncts of the original expression which cannot be unified with the consequent of any rule are simply added to the resulting regressed expression (with the substitutions applied to them). As illustrated in the figure, regressing the conjunct WEIGHT (x, w1) through the rule VOLUME (p1, v1) ∧ DENSITY (p1, d1) → WEIGHT (p1, v1*d1) therefore yields VOLUME (x, v1) ∧ DENSITY (x, d1). Regressing the conjunct WEIGHT (y, w2) through the rule ISA (p2, ENDTABLE) → WEIGHT (p2, 5) yields ISA (y, ENDTABLE). Finally, since no rule consequent can be unified with the conjunct LESS (w1, w2), this conjunct is simply added to the resulting regressed expression after applying the substitutions produced by regressing the other conjuncts. In this case these substitutions are {x/p1, v1*d1/w1, y/p2, 5/w2}, which yield the third conjunct LESS (v1*d1, 5). The final, operational definition for SAFE-TO-STACK (x, y) is therefore:

$$\begin{array}{ll} \text{VOLUME (x, v1)} & \\ \wedge\ \text{DENSITY (x, d1)} & \\ \wedge\ \text{LESS (v1*d1, 5)} & \\ \wedge\ \text{ISA (y, ENDTABLE)} & \rightarrow \text{SAFE-TO-STACK (x, y)} \end{array}$$

This expression characterizes in operational terms the features of the training example that are sufficient for the explanation structure to carry through in general. As such, it represents a justified generalization of the training example, for which the explanation structure serves as a justification.

2.3 Discussion

Several general points regarding the EBG method are illustrated in the above example. The main point of the above example is that the EBG method produces a justified generalization from a single training example in a two-step process. The first step creates an explanation that separates the relevant feature values in the examples from the irrelevant ones. The second step analyzes this explanation to determine the particular constraints on these feature values that are sufficient for the explanation structure to apply in general. Thus, explanation-based methods such as EBG overcome the main limitation of similarity-based methods: their inability to produce justified generalizations. This is accomplished by assuming that the learner has available knowledge of the domain, the goal concept, and the operationality

[6] It is correctly observed in DeJong (1986) that the substitution list used to regress expressions through previous steps in the explanation must be applied to the current expression before the next regression step.

criterion, whereas similarity-based generalization does not rely on any of these inputs.

A second point illustrated by the above example is that the language in which the final concept definition is stated can be quite rich. Notice in the above example that the final generalization includes a constraint that the product of the DENSITY and VOLUME of x must be less than 5. There are very many such relations among the parts of the training example that might be considered during generalization (e.g., why not consider the fact that the OWNERs of the two objects are of different SEX?). The interesting point here is that the appropriate constraint was derived directly by analyzing the explanation, without considering the universe of possible relations among parts of the training example. This is in marked contrast with similarity-based generalization methods (e.g. Michalski, 1983; Quinlan, 1985). Such methods are typically based on a heuristic focusing criterion, such as the heuristic that 'less complex features are preferred over more complex features for characterizing concepts'. Therefore, before such methods will consider the feature LESS (v1*d1, 5) as a plausible basis for generalization, they must first consider vast numbers of syntactically simpler, irrelevant features.

A final point illustrated by this example is that the final concept definition produced by EBG is typically a specialization of the goal concept rather than a direct reexpression of the concept. This is largely due to the fact that the explanation structure is created for the given training example, and does not explain every possible example of the goal concept. Thus, the generalization produced from analyzing this explanation will only cover examples for which the explanation holds. Furthermore, because the modified goal regression algorithm computes only sufficient (not necessary and sufficient) conditions under which the explanation holds, it leads to a further specialization of the concept. This limitation of explanation-based generalization suggests an interesting problem for further research: developing explanation-based methods that can utilize multiple training examples (see the discussion in Section 4).

3. Other examples and variations

This section discusses two additional examples of explanation-based generalization that have previously been reported in the literature. The first is Winston, Binford, Katz, and Lowry's (1983) research on learning structural definitions of concepts such as 'cup' from their functional definitions. The second is Mitchell, Keller, and Utgoff's (1983) research on learning search heuristics from examples (see also Utgoff, 1984; Keller, 1983). A common perspective on these two systems is provided by instantiating the EBG method for both problems. Differences among the two original approaches and the EBG method are also considered, in order to underscore some subtleties of the EBG method, and to suggest some possible variations.

3.1 An example: learning the concept CUP

In this subsection we first summarize the application of the EBG method to a second example of an explanation-based generalization problem: the CUP generalization problem, patterned after the work of Winston et al. (1983). We then discuss the relationship between the EBG method and the ANALOGY program of Winston et al., which addresses this same problem.

The CUP generalization problem, defined in Table 3, involves learning a structural definition of a cup from its functional definition. In particular, the goal concept here is the concept CUP, defined as the class of objects that are OPEN-VESSELs, LIFTABLE, and STABLE. The domain theory includes rules that relate these properties to the more primitive structural properties of physical objects, such as FLAT, HANDLE, etc. The operationality criterion requires that the output concept definition be useful for the task of visually recognizing examples of CUPs. It thus requires that the output concept definition be expressed in terms of its structural features.

Figure 3 illustrates a training example describing a particular cup, OBJ1, along with the explanation that shows how OBJ1 satisfies the goal concept CUP. In particular, the explanation indicates how OBJ1 is LIFTABLE, STABLE, and an OPEN-VESSEL. As in the SAFE-TO-STACK example, this explanation distinguishes the relevant features of the training example (e.g., its LIGHTness) from irrelevant features (e.g., its COLOR). The second step of the EBG method, regressing the goal concept through the explanation structure, results in the following general definition of the CUP concept:

$$\begin{aligned}&(\text{PART-OF}\,(x, xc) \wedge \text{ISA}\,(xc, \text{CONCAVITY}) \wedge \text{IS}\,(xc, \text{UPWARD-POINTING})\\ &\wedge \text{PART-OF}\,(x, xb) \wedge \text{ISA}\,(xb, \text{BOTTOM}) \wedge \text{IS}\,(xb, \text{FLAT})\\ &\wedge \text{PART-OF}\,(x, xh) \wedge \text{ISA}\,(xh, \text{HANDLE}) \wedge \text{IS}\,(x, \text{LIGHT})) \rightarrow \text{CUP}\,(x)\end{aligned}$$

*Table 3.*The CUP generalization problem after Winston et al. (1983)

Given:

- *Goal Concept:* Class of objects, x, such that CUP(x), where
  CUP(x) ⇔ LIFTABLE(x) ∧ STABLE(x) ∧ OPEN-VESSEL(x)
- *Training Example:*
  OWNER(OBJ1, EDGAR)
  PART-OF(OBJ1, CONCAVITY-1)
  IS(OBJ1, LIGHT)
  ...
- *Domain Theory:*
  IS(x, LIGHT) ∧ PART-OF(x, y) ∧ ISA(y, HANDLE) → LIFTABLE(x)
  PART-OF(x, y) ∧ ISA(y, BOTTOM) ∧ IS(y, FLAT) → STABLE(x)
  PART-OF(x, y) ∧ ISA(y, CONCAVITY) ∧ IS(y, UPWARD-POINTING → OPEN-VESSEL(x)
  ...
- *Operationality Criterion:* Concept definition must be expressed in terms of structural features used in describing examples (e.g., LIGHT, HANDLE, FLAT, etc.).

Determine:

- A generalization of training example that is a sufficient concept definition for the goal concept and that satisfies the operationality criterion.

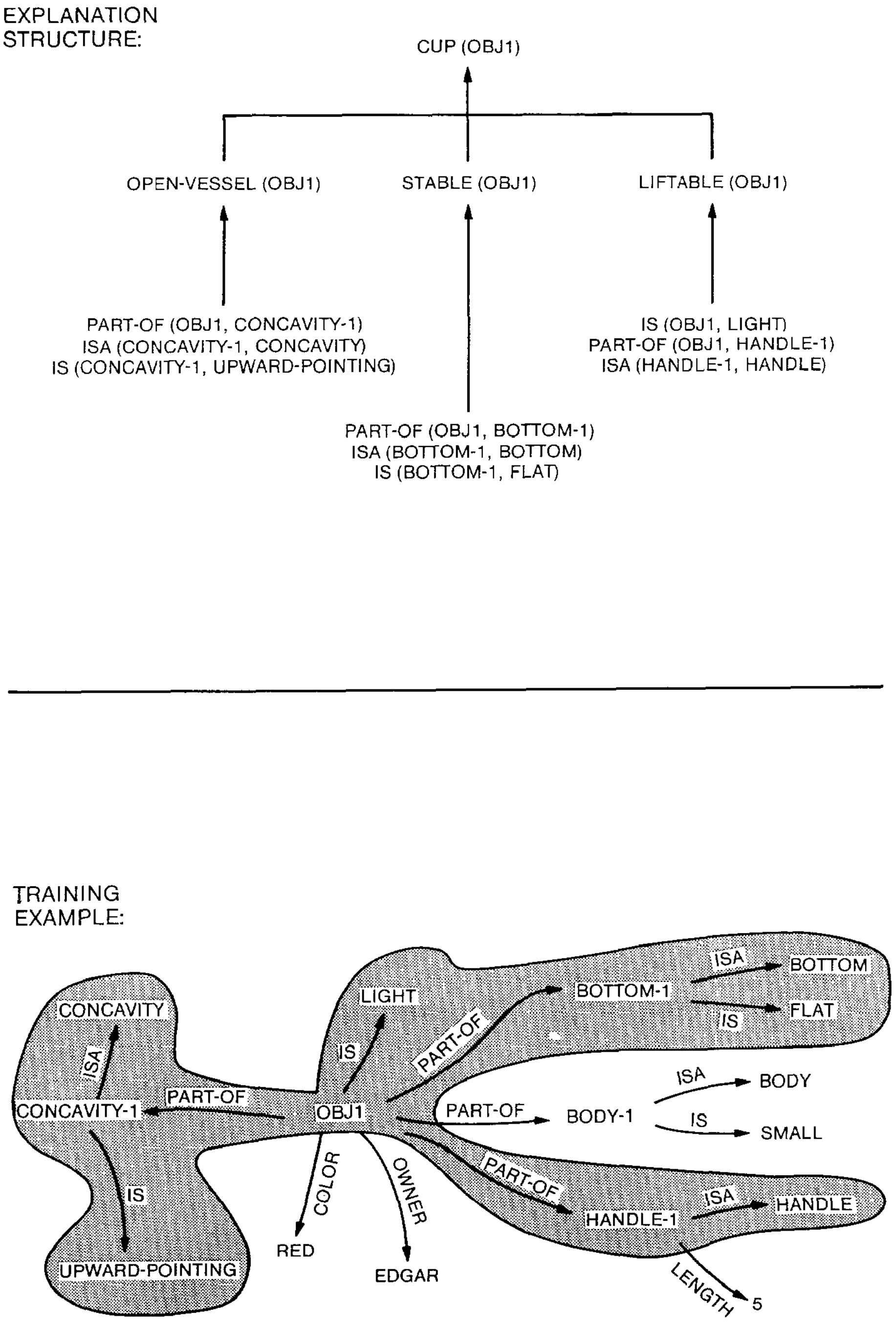

Figure 3. Explanation of CUP (OBJ1).

3.1.1 Discussion

As in the first example, the EBG method is able to produce a valid generalization from a single training example, by explaining and analyzing how the training example satisfies the definition of the goal concept. Notice that the regression step in this example is quite straightforward, and leads to generalizing the training example features effectively by replacing constants with variables (i.e., replacing OBJ1 by x). It is interesting that several earlier attempts at explanation-based generalization (e.g., Mitchell, 1983) involved the assumption that the explanation could always be generalized simply by replacing constants by variables in the explanation, without the need to regress the goal concept through the explanation structure.[7] As the earlier SAFE-TO-STACK example illustrates, this is not the case. In general, one must regress the goal concept through the explanation structure to ensure a valid generalization of the training example (which may involve composing terms from different parts of the explanation, requiring constants where no generalization is possible, and so on).

Several interesting features of this example come to light when the EBG method is compared to the method used in Winston et al.'s (1983) ANALOGY program, upon which this example problem is based. The most striking difference between the two methods is that although ANALOGY does construct an explanation, and also uses this explanation to generalize from a single example, the system has no domain theory of the kind used in the above example. Instead, Winston's program constructs its explanations by drawing analogies between the training example and a library of precedent cases (e.g., annotated descriptions of example suitcases, bricks, bowls, etc.). For example, ANALOGY explains that the FLAT BOTTOM of OBJ1 allows OBJ1 to be STABLE, by drawing an analogy to a stored description of a brick which has been annotated with the assertion that its FLAT BOTTOM 'causes' it to be STABLE. Similarly, it relies on an annotated example of a suitcase to explain by analogy why a handle allows OBJ1 to be LIFTABLE.

In general, the precedents used by ANALOGY are assumed to be annotated by links that indicate which features of the precedent account for which of its properties. Thus, for the ANALOGY program, the causal links distributed over the library of precedents constitute its domain theory. However, this theory is qualitatively different than the domain theory used in the CUP example above: it is described by extension rather than intention (i.e., by examples rather than by general rules), and is therefore a weaker domain theory. Because ANALOGY's knowledge about causality is summarized by a collection of instances of causal relations rather than by general rules of causality, its theory cannot lead *deductively to* assertions about new causal links.

[7] This observation is due in part to Sridhar Mahadevan.

Because its domain theory is weak, the ANALOGY system raises some interesting questions about explanation-based generalization. Whereas the SAFE-TO-STACK and CUP examples above show how a sufficient set of domain theory rules can provide powerful guidance for generalization, ANALOGY suggests how a weaker, extensional theory might be used to focus the generalization process in a weaker fashion. In particular, the causal links are used by ANALOGY to construct a plausible explanation, but not a proof, that the training example satisfies the goal concept definition. As discussed above, such plausible explanations can guide generalization by focusing on plausibly relevant features of the training example. But since ANALOGY lacks general inference rules to characterize the links in this explanation, it cannot perform the second (goal regression) step in the EBG method, and therefore has no valid basis for generalizing the explanation. In fact, the ANALOGY program generalizes anyway, implicitly, assuming that each causal link of the form (feature (OBJ1) → property (OBJ1)) is supported by a general rule of the form (($\forall$x) feature (x) → property (x)).[8] Thus, ANALOGY represents an important step in considering the use of a weak domain theory to guide generalization, and helps to illuminate a number of open research issues (see the discussion in Section 4).

3.2 *An example: learning a search heuristic*

This section presents a third example of an explanation-based generalization problem – this one involving the learning of search heuristics. This example is based on the problem addressed by the LEX program (Mitchell et al., 1983) which learns search control heuristics for solving problems in the integral calculus. In particular, LEX begins with a set of legal operators (transformations) for solving integrals (e.g., integration by parts, moving constants outside the integrand). For each such operator, the system learns a heuristic that summarizes the class of integrals (problem states) for which it is useful to apply that operator. For example, one typical heuristic learned by the LEX system is:

IF the integral is of the form $\int$ *<polynomial–fn>·<trigonometric–fn>dx*,
THEN apply Integration-by-Parts

Thus, for each of its given operators, LEX faces a generalization problem: learning the class of integrals for which that operator is useful in reaching a solution.

Table 4 defines the generalization problem that corresponds to learning when it is useful to apply OP3 (moving constants outside the integrand). Here the goal concept USEFUL-OP3 (x) describes the class of problem states (integrals) for which OP3 will

[8] Winston's (1985) own work on building rule censors can be viewed as an attempt to address difficulties that arise from this implicit assumption.

Table 4. The search heuristic generalization problem after Mitchell et al. (1983)

Given:

- *Goal Concept:* The class of integral expressions that can be solved by first applying operator OP3 (removing a constant from the integrand); that is, the class of integrals, x, such that USEFUL-OP3 (x), where

  USEFUL-OP3(x) $\Leftrightarrow$ NOT(SOLVED(x)) $\wedge$ SOLVABLE(OP3(x))

  and

  OP3: $\int r \cdot {<}\mathit{any-fn}{>} dx \rightarrow r \int {<}\mathit{any-fn}{>} dx$.
- *Training Example:* $\int 7x^2 dx$
- *Domain Theory:*
  SOLVABLE(x) $\Leftrightarrow$ ($\exists$op) (SOLVED(op(x)) $\vee$ SOLVABLE(op(x)))
  MATCHES(x, '$\int {<}\mathit{any-fn}{>} dx$') $\rightarrow$ NOT (SOLVED(x))
  MATCHES(x, '${<}\mathit{any-fn}{>}$') $\rightarrow$ SOLVED(x)
  MATCHES(op(x), y) $\Leftrightarrow$ MATCHES(x, REGRESS(y, op))
- *Operationality Criterion:* Concept definition must be expressed in a form that uses *easily-computable* features of the problem state, x, (e.g., features such as <polynomial-fn>, <transcendental-fn>, <any-fn>, r, k).

Determine:

- A generalization of training example that is a sufficient concept definition of the goal concept and that satisfies the operationality criterion.

be useful. This is defined to be the class of problem states that are NOT already SOLVED (i.e., algebraic expressions that contain an integral sign), and for which applying OP3 leads to a SOLVABLE problem state. The domain theory in this case contains rules that relate SOLVED and SOLVABLE to observable features of problem states (e.g., one rule states that if problem state x MATCHES the expression $\int {<}\mathit{any-fn}{>} dx$, then x is NOT a SOLVED state.). Notice that some of the domain theory rules constitute knowledge about search and problem solving in general (e.g., the definition of SOLVABLE), while other rules represent knowledge specific to integral calculus (e.g., that the absence of an integral sign denotes a SOLVED state). The operationality condition in this problem requires that the final concept definition be stated in terms of *easily-computable* features of the given problem state. This requirement assures that the final concept definition will be in a form that permits its effective use as a search control heuristic.[9] For the LEX program, the set of *easily-computable* features is described by a well-defined generalization language over problem states that includes features (e.g., <trigonometric-fn>, <real-constant>) which LEX can efficiently recognize using its MATCHES predicate.

[9] If the concept definition were permitted to include features that are difficult to compute (e.g., SOLVABLE), then the resulting heuristic would be so expensive to evaluate that its use would degrade, rather than improve, the problem solver's performance.

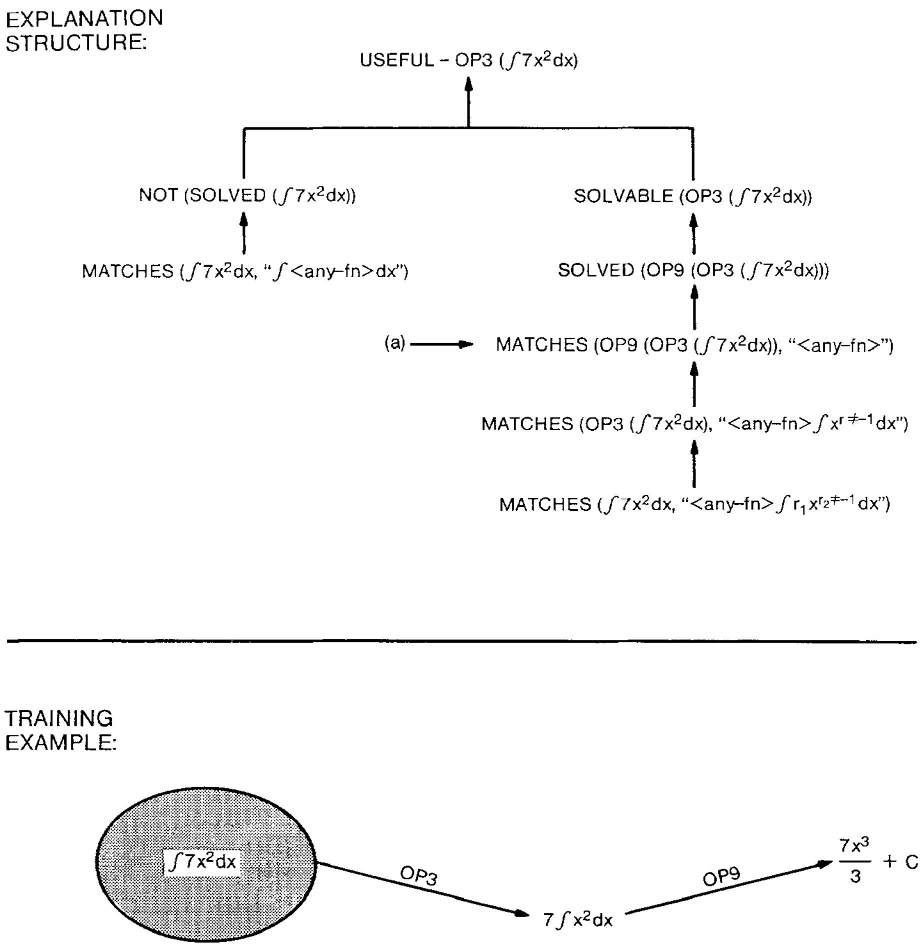
EXPLANATION STRUCTURE:
USEFUL - OP3 ($\int 7x^2dx$)
NOT (SOLVED ($\int 7x^2dx$))
MATCHES ($\int 7x^2dx$, "$\int$<any-fn>dx")
SOLVABLE (OP3 ($\int 7x^2dx$))
SOLVED (OP9 (OP3 ($\int 7x^2dx$)))
(a)
MATCHES (OP9 (OP3 ($\int 7x^2dx$)), "<any-fn>")
MATCHES (OP3 ($\int 7x^2dx$), "<any-fn>$\int x^{r \neq -1}dx$")
MATCHES ($\int 7x^2dx$, "<any-fn>$\int r_1 x^{r_2 \neq -1}dx$")
TRAINING EXAMPLE:
$\int 7x^2dx$
OP3
$7\int x^2dx$
OP9
$\frac{7x^3}{3} + C$

Figure 4. Explanation of USEFUL-OP3 ($\int 7x^2 dx$).

A training example for the goal concept USEFUL-OP3 is shown in the bottom portion of Figure 4. In particular, the problem state $\int 7x^2 dx$ constitutes a training example of a problem state for which application of OP3 is useful. The training example described in the figure is shown along with the other problem states involved in its solution.

The explanation of USEFUL-OP3 ($\int 7x^2 dx$) is shown in the top portion of Figure 4. The left-hand branch of this explanation structure leads to a node which asserts that USEFUL-OP3 is satisfied in part because the training example state is not already a SOLVED state. The right-hand branch of the explanation structure explains that applying OP3 to the example state leads to a SOLVABLE problem state.

This is in turn explained by indicating that applying OP9[10] to the resulting state produces a SOLVED problem, as evidenced by the fact that the resulting state MATCHES the expression '$<any-fn>$' (i.e., that it contains no integral sign). Thus, up to this point (marked as (a) in the figure), each step in the right-hand branch of the explanation structure corresponds to some step along the solution path of the training example.

By point (a), the explanation has indicated that one relevant feature of the training example state is that the result of applying OP3 followed by OP9, is a state that MATCHES '$<any-fn>$'. The operationality criterion requires, however, that the explanation be in terms of features of the single given training example state, rather than features of its resulting solution state. Thus, the remainder of the explanation consists of reexpressing this constraint in terms of the training example state. This is accomplished by applying the last rule in the domain theory of Table 4. This rule[11] allows back propagating the expression '$<any-fn>$' through the general definitions of OP9 and OP3, to determine the equivalent constraint on the training example state. The resulting constraint is that the training example state must MATCH the expression '$<any-fn> \int r_1 \cdot x\, r_2 \neq -1\, dx$' (Here r_1 and r_2 stand for two distinct real numbers, where r_2 must not be equal to -1.). This together with the left-hand branch of the explanation structure, explains which features of $\int 7x^2 dx$ guarantee that it satisfies the goal concept USEFUL-OP3.

Given this explanation structure, the second step of the EBG method is straightforward. As in the CUP example, regressing the goal concept expression USEFUL-OP3 (x) through the explanation structure effectively results in replacing the training example state by a variable, so that the resulting generalization (taken from the leaves of the explanation tree) is:

$$\begin{aligned}&\text{MATCHES (x, `}\textstyle\int <any-fn>dx\text{')} \wedge \\ &\qquad\qquad \text{MATCHES (x, `}<any-fn> \textstyle\int r_1 \cdot x r_2 \neq -1\, dx\text{')} \rightarrow \text{USEFUL-OP3}\end{aligned}$$

which simplifies to:

$$\text{MATCHES (x, `}<any-fn> \textstyle\int r_1 \cdot x r_2 \neq -1\, dx\text{')} \rightarrow \text{USEFUL-OP3 (x)}$$

[10] OP9: $\int x^{r \neq -1} dx \rightarrow x^{r+1}/(r+1)$.

[11] The domain theory rule MATCHES (op (x), y) ⇔ MATCHES (x, REGRESS (y, op)) indicates that if the result of applying operator *op* to state *x* MATCHES some expression *y*, then the state *x* MATCHES some expression which can be computed by REGRESSing the expression *y* through operator *op*. Notice that the regression here involves propagating constraints on problem states through problem solving operators. This is a different regression step from the second step of the EBG process, in which the goal concept is regressed through the domain theory rules used in the explanation structure.

3.2.1 Discussion

To summarize, this example demonstrates again the general EBG method of constructing an explanation in terms that satisfy the operationality condition, then regressing the goal concept through the explanation structure to determine a justified generalization of the training example. As in the previous examples, this process results in a generalization of the training example which is a sufficient condition for satisfying the goal concept, and which is justified in terms of the goal concept, domain theory, and operationality criterion.

In the above example, the goal concept corresponds to the precondition for a search heuristic that is to be learned. The domain theory therefore involves both domain-independent knowledge about search (e.g., the definition of SOLVABLE) and domain-dependent knowledge (e.g., how to recognize a SOLVED integral). To use this method to learn heuristics in a new domain, one would have to replace only the domain-dependent portion of the theory. To learn a different type of heuristic in the same domain, one could leave the domain theory intact, changing only the definition of the USEFUL-OP3 goal concept accordingly. For example, as suggested in Mitchell (1984), the system could be modified to learn heuristics that suggest only steps along the *minimum cost* solution path, by changing the goal concept to

$$\begin{aligned}\text{USEFUL-OP3 (s)} \Leftrightarrow\ & \text{NOT(SOLVED (s))} \wedge \\ & \text{MIN-COST-SOLN (SOLUTION-PATH (OP3, s))}\end{aligned}$$

Note that the explanation structure in this example, like the domain theory, separates into a domain-independent and a domain-dependent portion. Domain-independent knowledge about search is applied above point (a) in Figure 4, and domain-dependent knowledge about calculus problem solving operators is applied below point (a). In the implementation of the LEX2 program (Mitchell, 1983), these two phases of the explanation were considered to be two unrelated subprocesses and were implemented as separate procedures. From the perspective afforded by the EBG method, however, these two subprocesses are better seen as different portions of the same explanation-generation step.

One final point regarding the current example has to do with the ability of explanation-based methods to augment their description language of concepts. In LEX2, as in the SAFE-TO-STACK problem, this method is able to isolate fairly complex features of the training example that are directly related to the explanation of how it satisfies the goal concept. In the context of the LEX project, Utgoff (1985) studied this issue and developed the STABB subsystem. STABB is able to extend the initial vocabulary of terms used by LEX, by naming and assimilating terms that correspond to the constraints derived during the regression step. For example, in one instance STABB derived the definition of odd integers through this regression step, defining it as 'the set of real numbers, x, such that subtracting 1 then dividing by 2 produces an integer'.

3.2.2 Related methods for strategy learning

There are a number of additional systems that learn problem solving strategies by analyzing single examples of successful solutions.[12] These systems (e.g. Fikes et al., 1972; Utgoff, 1984; Minton, 1984; Mahadevan, 1985), which we might call STRIPS-like systems, can all be viewed as systems that learn a goal concept of the following form: 'the set of problem states such that applying a given operator sequence, OS, yields a final state matching a given solution property, P.' Since these systems are tuned to this single goal concept, and are not intended to learn other forms of concepts, they typically do not represent the goal concept and explanation declaratively. However, they do represent the solution property, P, explicitly, and regress this property through the operator sequence OS to determine an operational definition of the (implicit) goal concept. From the perspective of the above LEX example, the steps that they perform correspond to constructing the portion of the explanation below point (a) in Figure 4. It is in constructing these steps of the explanation that LEX regresses its solution property 'MATCHES (x, <*any*−*fn*>)' through the operator sequence <OP3, OP9> to determine the equivalent constraint on the initial problem state. These STRIPS-like systems do not construct the portion of the explanation corresponding to the section above point (a) in Figure 4. Because they are tuned to a fixed goal concept, they do not need to generate this portion of the explanation explicitly for each training example.

To illustrate this point, consider Minton's (1984) program for learning search heuristics in two-person games such as Go-Moku, Tic-Tac-Toe, and Chess. This program analyzes one positive instance of a sequence of moves that leads to a winning board position, in order to determine an operational definition of the goal concept 'the class of board positions for which the given sequence of moves leads to a forced win'. But Minton's system has no explicit definition of this goal concept. It has only a definition of the solution property P that characterizes a winning position. For example, in the game of Go-Moku, (a variant of Tic-Tac-Toe) this solution property characterizes a winning board position as 'a board position with five X's in a row'. This solution property is regressed by Minton's program through the operator sequence for the given training example. In this way, the program determines that an effective definition of its implicit goal concept is 'board positions that contain three X's, with a blank space on one side, and two blank spaces on the other'

[12] See Kibler and Porter (1985) for a thoughtful critique of analytic goal regression methods for learning search control heuristics. They discuss certain requirements for regression to succeed: that the operators be invertible, and that the representation language be able to express goal regression products.

4. Perspective and research issues

The previous sections presented a general method for explanation-based generalization, and illustrated its application to several generalization tasks. This section summarizes some general points regarding explanation-based generalization, and considers a number of outstanding research issues.

To summarize, explanation-based generalization utilizes a domain theory and knowledge of the goal concept to guide the generalization process. By doing so, the method is able to produce a valid generalization of the training example, along with an explanation that serves as a justification for the generalization. The EBG method introduced here unifies mechanisms for explanation-based generalization that have been previously reported for a variety of task domains. The generality of the EBG method stems from the fact that the goal concept, domain theory, and operationality criterion are made explicit inputs to the method, rather than instantiated implicitly within the method.

4.1 Perspectives on explanation-based generalization

Several perspectives on explanation-based generalization, and on the EBG method in particular, are useful in understanding their strengths and weaknesses:

EBG as theory-guided generalization of training examples. EBG can be seen as the process of interpreting or perceiving a given training example as a member of the goal concept, based on a theory of the domain. Soloway's (1978) early work on learning action sequences in the game of baseball shares this viewpoint on generalization. This is the perspective stressed in the above sections, and it highlights the centrality of the goal concept and domain theory. It also highlights an important feature of EBG: that learning depends strongly on what the learner already knows. One consequence of this is that the degree of generalization produced for a particular training example will depend strongly on the generality with which the rules in the domain theory are expressed. A second consequence is that the learning system can improve its *learning* performance to the degree that it can learn new rules for its domain theory.

EBG as example-guided operationalization of the goal concept. One can also view EBG as the process of reformulating the goal concept in terms that satisfy the operationality criterion, with the domain theory providing the means for reexpressing the goal concept. Given this perspective, one wonders why training examples are required at all. In principle, they are not. Mostow's (1983) FOO system operationalizes general advice about how to play the card game of Hearts, without considering specific examples that apply that advice. Similarly, Keller (1983) describes a process of *concept operationalization*, by which a sequence of transformations is applied to the goal concept in search of a reformulation that satisfies the operationality criterion, without the guidance of specific training examples.

However, training examples can be critical in guiding the learner to consider relevant transformations of the goal concept. For instance, consider the CUP learning task as described in Section 3.1, where a functional definition of CUP is reexpressed in structural terms for use by a vision system recognizing cups. A system that reformulates the functional definition of CUP in structural terms, without the guidance of training examples, amounts to a system for producing all possible structural definitions for classes of cups (i.e., for *designing* all possible classes of cups). Since there are so many possible designs for cups, and since so few of these are actually encountered in the world, the learning system could easily waste its effort learning structural definitions corresponding to cups that will never be seen by the vision system![13] Training examples thus provide a means of focusing the learner on formulating only concept descriptions that are relevant to the environment in which it operates.

EBG as Reformulating/Operationalizing/Deducing from what is already known. The above paragraph suggests that explanation-based generalization does not lead to acquiring truly 'new' knowledge, but only enables the learner to reformulate/operationalize/deduce what the learner already knows implicitly. While this statement is true, it is somewhat misleading. Consider, for example, the task of learning to play chess. Once one is told the rules of the game (e.g., the legal moves, and how to recognize a checkmate), one knows *in principle* everything there is to know about chess – even the optimal strategy for playing chess follows deductively from the rules of the game. Thus, although the EBG method is restricted to compiling the deductive consequences of its existing domain theory, this kind of learning is often nontrivial (as is the case for learning chess strategies). Nevertheless, it is a significant limitation that EBG is highly dependent upon its domain theory. As discussed below, further research is needed to extend the method to generalization tasks in which the domain theory is not sufficient to deductively infer the desired concept.

4.2 Research issues

4.2.1 Imperfect theory problems

As the above discussion points out, one important assumption of EBG is that the

[13] Of course information about what types of cups are to be encountered by the vision system also could be presented in the *operationality criterion*, since this information relates to the *use* of the concept definition for the recognition task. This information, however, may not be easily described in the declarative form required by the *operationality criterion*.

domain theory is sufficient to *prove* that the training example is a member of the goal concept; that is, that the inferred generalizations follow deductively (even if remotely) from what the learner already knows. Although this assumption is satisfied in each of the example problems presented above, and although there are interesting domains in which this assumption is satisfied (e.g., chess, circuit design (Mitchell et al., 1985)), for the majority of real-world learning tasks it is unrealistic to assume that the learner begins with such a strong theory. For both the SAFE-TO-STACK domain and the CUP domain, it is easy to imagine more realistic examples for which the required domain theory is extremely complex, difficult to describe, or simply unknown. For generalization problems such as inferring general rules for predicting the stock market or the weather, it is clear that available theories of economics and meteorology are insufficient to produce absolutely predictive rules. Thus, a major research issue for explanation-based generalization is to develop methods that utilize imperfect domain theories to guide generalization, as well as methods for improving imperfect theories as learning proceeds. The problem of dealing with imperfect theories can be broken down into several classes of problems:

The Incomplete Theory Problem. The stock market and weather prediction examples above both illustrate the incomplete theory problem. The issue here is that such theories are not complete enough to *prove* that the training example is a member of the goal concept (e.g., to prove why a particular training example stock has doubled over a twelve month period). However, even an incomplete theory might allow constructing *plausible explanations* summarizing likely links between features of the training example and the goal concept. For example, even a weak theory of economics allows one to suggest that the 'cash on hand' of the company may be relevant to the goal concept 'stocks that double over a twelve month period', whereas the 'middle initial of the company president' is probably an irrelevant feature. Thus, incomplete theories that contain only information about plausible cause-effect relations, with only qualitative rather than quantitative associations, can still provide important guidance in generalizing. Methods for utilizing and refining such incomplete theories would constitute a major step forward in understanding explanation-based generalization.

The Intractable Theory Problem. A second class of imperfect theories includes those which are complete, but for which it is computationally prohibitive to construct explanations in terms of the theory. For instance, quantum physics constitutes a fairly complete theory that would be inappropriate for generating explanations in the SAFE-TO-STACK problem – generating the necessary explanations in terms of quantum physics is clearly intractable. Similarly, although the rules of chess constitute a domain theory sufficient to explain why any given move is good or bad, one would never use this theory to explain why the opening move 'pawn to king four' is a member of the goal concept 'moves that lead to a win or draw for white'. In fact, this theory of chess is intractable for explaining why nearly any move is good or bad. Humans tend to respond to this problem by constructing more abstract, tractable theories that are approximations to the underlying intractable theory. In chess, for example, the learner might formulate a more abstract theory that includes approximate assertions such as 'there is no threat to the king if it is surrounded by many friendly pieces' (Tadepalli, 1985). Such approximate, abstracted theories can be tractable enough and accurate enough to serve as a useful basis for creating and learning from explanations. Developing computer methods that can construct such abstracted theories, and that can judge when they can safely be applied, is a problem for further research.

The Inconsistent Theory Problem. A third difficulty arises in theories from which inconsistent statements can be derived. The domain theory in the SAFE-TO-STACK problem provides one example of such a theory. While this theory has a default rule for inferring the weight of an end table, it also has a rule for computing weights from the known density and volume. Thus, the theory will conclude two different weights for a given end table provided that its density and volume are known, and provided that these are inconsistent with the default assumption about its weight. In such cases, it is possible to construct inconsistent explanations for a single training example. Furthermore, if two different training examples of the same concept are explained in inconsistent terms (e.g., by utilizing one default assumption for one example, and some other assumptions for the second example), difficulties will certainly arise in merging the resulting generalizations. Because of this, and because default assumptions are commonplace in theories of many domains, the problem of dealing with inconsistent theories and inconsistent explanations is also an important one for future research.

4.2.2 Combining explanation-based and similarity-based methods

While EBG infers concept definitions deductively from a single example, similarity-based methods infer concept definitions inductively from a number of training examples. It seems clearly desirable to develop combined methods that would utilize both a domain theory and multiple training examples to infer concept definitions. This kind of combined approach to generalization will probably be essential in domains where only imperfect theories are available.

Although few results have been achieved in combining explanation-based and similarity-based methods, a number of researchers have begun to consider this issue. Lebowitz (Lebowitz, 1985) is exploring methods for combining similarity-based methods and explanation-based methods in his UNIMEM system. UNIMEM examines a database of the voting records of congresspersons, searching for empirical, similarity-based generalizations (e.g., midwestern congresspersons vote in favor of farm subsidies). The system then attempts to verify these empirical generalizations by explaining them in terms of a domain theory (e.g. explaining how midwestern congresspersons satisfy the goal concept 'people who favor farm subsidies'). This approach has the advantage that the similarity-based techniques can be used to generate a candidate set of possible generalizations from a large number of potentially noisy training examples. Once such empirical generalizations are formulated, explanation-based methods can help prune and refine them by using other knowledge in the system.

Whereas Lebowitz's approach involves applying similarity-based generalization followed by explanation-based methods, an alternative approach is to first apply explanation-based methods to each training example, then to combine the resulting generalized examples using a similarity-based generalization technique. Consider, for example, using the version space method[14] to combine the results of explanation-based generalizations of a number of training examples (Mitchell, 1984). Since the explanation-based generalization of a positive training example constitutes a sufficient condition for the goal concept, this can be used as a generalized positive example to refine (generalize) the specific boundary set of the version space. Similarly, one could imagine generalizing negative training examples using explanation-based generalization, by explaining why they are not members of the goal concept. The resulting generalized negative example could then be used to refine (specialize) the general boundary set of the version space. Thus, while this combined method still suffers the main disadvantage of similarity-based methods (i.e., it makes inductive leaps based on its generalization language, which it cannot justify), it converges more rapidly on a final concept definition because it employs EBG to generalize each training example.

Kedar-Cabelli (1984, 1985) proposes an alternative method for combining the results of explanation-based generalizations from multiple training examples. This method, Purpose-Directed Analogy, involves constructing an explanation of one example by analogy with an explanation of a familiar example, then combining the two explanations to produce a general concept definition based on both. Given explanations for two different examples, the proposed system combines the explanations as follows: Common portions of the two explanations remain unaltered in the combined explanation. Differing portions either become disjunctive subexpressions in the combined explanation, or are generalized to the next more specific common subexpression in the explanation. For example, given an explanation that a blue, ceramic mug is a CUP, and a second example of a white styrofoam cup, the explanation of the first example is used to construct by analogy an explanation for the

The version space method (Mitchell, 1978) is a similarity-based generalization method based on summarizing the alternative plausible concept definitions by maintaining two sets: the 'specific' set contains the set of most specific concept definitions consistent with the observed data, and the 'general' set contains the most general concept definitions consistent with the data. All other plausible concept definitions lie between these two sets in the general-to-specific ordering over concept definitions.

second example. The two resulting explanations may differ in how they explain that the two example cups are GRASPABLE (assume the first example cup is GRASPABLE because it has a handle, whereas the second is GRASPABLE because it is conical). In this case, a generalization of the two explanations would include a disjunction, that either the conical shape, or a handle, makes it graspable. That, along with the common features of the two objects in the combined explanation structure leads to the generalization that cups include objects which are concave upward, have a flat bottom, are light, and have either a conical shape or a handle. Alternatively, the combined explanation would retain only the next most-specific common subexpression, GRASPABLE, which would lead to a slightly more general, yet less operational, definition of a cup. Thus, this method of combining explanations of multiple examples provides a principled method for introducing disjunctions where needed into the common generalization of the two examples.

The three methods discussed above for combining similarity-based and explanation-based generalization offer differing advantages. The first method uses similarity-based generalization to determine empirical generalizations which may then be validated and refined by explanation-based methods. The second method involves employing a similarity-based method to combine the results of explanation-based generalizations from multiple examples. It suffers the disadvantage that this combination of methods still produces unjustified generalizations. The third method merges the explanations of multiple examples in order to produce a combined generalization that is justified in terms of the merged explanations. More research is required on these and other possible methods for employing explanation-based methods when multiple training examples are available.

4.2.3 Formulating generalization tasks

The above discussion focuses on research issues within the framework of explanation-based generalization. An equally important set of research issues has to do with how such methods for generalization will be used as subcomponents of larger systems that improve their performance at some given task. As our understanding of generalization methods advances, questions about how to construct performance systems that incorporate generalization mechanisms will become increasingly important.

One key issue to consider in this regard is how generalization tasks are initially formulated. In other words, where do the inputs to the EBG method (the goal concept, the domain theory, the operationality criterion) come from? Is it possible to build a system that automatically formulates its own generalization tasks and these inputs? Is it possible to build learning systems that automatically shift their focus of attention from one learning problem to the next as required? What kind of knowledge must be transmitted between the performance system and the learning system to enable the automatic formulation of generalization tasks?

Again, little work has been devoted to these issues. The SOAR system (Laird et al., 1984, Laird et al., 1986) is one example of a learning system that formulates its own generalization tasks. Each time that SOAR encounters and solves a subgoal, it formulates the generalization problem of inferring the general conditions under which it can reuse the solution to this subgoal. SOAR then utilizes a technique closely related to explanation-based generalization, called *implicit generalization* (Laird et al., 1986), to infer these subgoal preconditions.

A second research effort which confronts the problem of formulating learning tasks is Keller's research on *contextual learning* (Keller, 1983, 1985, 1986). In this work, Keller suggests how a problem solving system could itself formulate generalization problems such as those addressed by the LEX2 system. In particular, he shows how the task of learning the goal concept USEFUL-OP3 arises as a subgoal in the process of planning to improve performance at solving calculus problems. By reasoning from a top-level goal of improving the efficiency of the problem solver, as well as a declarative description of the problem solving search schema, the method derives the subgoal of introducing a filter to prune the search moves that it considers. The definition of this filter includes the specification that it is to allow only problem solving steps that are 'useful' (i.e., that lead toward solutions). The subgoal of introducing this filter leads, in turn, to the problem of operationalizing the definition of 'useful' (i.e., to the subgoal corresponding to the LEX2 generalization task).

Recent work by Kedar-Cabelli (1985) also addresses the problem of formulating learning tasks. In this work, Kedar-Cabelli proposes a system to automatically formulate definitions of goal concepts in the domain of artifacts. In particular, the proposed system derives functional definitions of artifacts (e.g., CUP) from information about the purpose for which agents use them (e.g., to satisfy their thirst). Given two different purposes for which an agent might use a cup (e.g., as an ornament, versus to satisfy thirst), two different functional definitions can be derived.[15] To derive the functional definition of the artifact, the proposed system first computes a plan of actions that leads to satisfying the agent's goal. For example, if the agent's goal is to satisfy thirst, then this plan might be to POUR the liquids into the cup, GRASP the cup with the liquid in order to LIFT, and finally DRINK the liquids. In order to be used as part of this plan, the artifact must satisfy the preconditions of those plan actions in which it is involved. These preconditions form the functional definition of a cup: an open-vessel, which is stable, graspable, liftable. Thus, formulating functional definitions of artifacts is accomplished by analyzing the role that the artifact plays in facilitating the goal of some agent.

[15] This extends Winston's work (see Section 3.1), in that it can derive its own goal concept from a given purpose.

4.2.4 Using contextual knowledge to solve the generalization task

Above we have discussed some approaches to automatically formulating learning tasks, given knowledge of the performance task for which the learning takes place. In cases where the learner formulates its own learning task, information about how and why the task was formulated can provide important guidance in solving the learning task. Keller's (1986) METALEX system provides an example of how such information can be used in guiding learning. Like LEX2, METALEX addresses the learning task of operationalizing the goal concept USEFUL-OP3. It takes as input a procedural representation of the *performance task* to be improved (the calculus problem solver), a specification of the *performance objectives* to be achieved ('minimize problem solving time') and knowledge of the *performance improvement plan* (search space pruning via filtering), which is a record of how the operationalization task was originally formulated. METALEX uses this additional knowledge about the context in which its learning task was formulated to guide its search for an operational transformation of the goal concept. Specifically, it executes the calculus problem solver using the initial (and subsequent intermediary) definitions of the goal concept to collect diagnostic information which aids in operationalizing the goal concept if performance objectives are not satisfied.

In effect, the performance task and performance objective inputs required by METALEX elaborate on the operationality criterion required by the EBG method. Instead of evaluating operationality in terms of a binary-valued predicate over concept definitions (as in EBG), METALEX evaluates the *degree* of operationality of the concept definition in relation to the performance task and objective. This ability to make a more sophisticated analysis of operationality enables METALEX to make important distinctions among alternative concept definitions. For example, because METALEX uses approximating (non truth-preserving) transforms to modify the goal concept, it can generate concept definitions that only approximate the goal concept. In such cases, METALEX is able to determine whether such an approximate concept definition is desirable based on the degree to which it helps improve the performance objectives.

As the first sections of this paper demonstrate, explanation-based generalization methods offer significant promise in attempts to build computer models of learning systems. Significant progress has been made in understanding explanation-based generalization, especially for problems in which the learner possesses a complete and correct theory. As the final section illustrates, much more remains to be discovered about how a learner can use what it already knows to guide the acquisition of new knowledge.

Acknowledgments

The perspective on Explanation-Based Generalization reported here has arisen from discussions over a period of time with a number of people in the Machine Learning Group at Rutgers and elsewhere. Discussions with the following people have been particularly useful in formulating the ideas presented here: Alex Borgida, Gerry DeJong, Thomas Dietterich, Thorne McCarty, Sridhar Mahadevan, Jack Mostow, Michael Sims, Prasad Tadepalli, Paul Utgoff, and Keith Williamson. We also thank the following people for providing useful comments on an earlier draft of this paper: Pat Langley, Sridhar Mahadevan, Jack Mostow, Louis Steinberg, Prasad Tadepalli, and Keith Williamson.

This material is based on work supported by the Defense Advanced Research Projects Agency under Research Contract N00014-81-K-0394, by the National Science Foundation under grants DCS83-51523 and MCS80-08889, by the National Institutes of Health under grant RR-64309, by GTE under grant GTE840917, and by a Rutgers University Graduate Fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official views or policies of these sponsors.

Appendix

The appendix describes DeJong's research on explanation-based generalization. In particular, it casts the work on learning schemata for story understanding in terms of the EBG method. In addition to this project, there has been a great deal of recent research on explanation-based generalization, including (DeJong, 1985; Ellman, 1985; Mooney, 1985; O'Rorke, 1985; Rajamoney, 1985; Schooley, 1985; Segre, 1985; Shavlik, 1985; Sims, 1985; Watanabe, 1985; Williamson, 1985).

DeJong (1981, 1983) developed one of the earliest successful explanation-based generalization systems as part of his research on *explanatory schema acquisition*. DeJong is interested in the problem of learning schemata for use in natural language story understanding. DeJong's system takes as input an example story and produces as output a generalized schema representing the stereotypical action sequence that is instantiated in the story. For example, the system can process the following story (adapted from G. DeJong, personal communication, November 16, 1984):

> Fred is Mary's father. Fred is rich. Mary wears blue jeans. John approached Mary. He pointed a gun at her. He told her to get into his car. John drove Mary to the hotel. He locked her in his room. John called Fred. He told Fred he had Mary. He promised not to harm her if Fred gave him $250,000 at Treno's Restaurant. Fred delivered the money. Mary arrived home in a taxi.

It then produces as output a generalized schema for KIDNAPPING. The KIDNAPPING schema contains only the relevant details of the kidnapping (e.g., that three people are involved: Person A who wants money, Person B who has money and Person C who is valued by Person B), but none of the irrelevant details (e.g., that Person C wears blue jeans).

DeJong's system uses a generalization method that closely parallels the EBG method. Although there is no direct counterpart to the goal concept in DeJong's system, the goal concept can be thought of as 'the class of action sequences that achieve personal goal X for actor Y.' For the kidnapping story, the actor's personal goal is 'attainment of wealth.' The system constructs an explanation for how the actions in the story lead to the kidnapper's 'attainment of wealth' as a by-product of the story-understanding process. During the story parse, *data dependency links* are created to connect actions in the story with the inference rules that are used by the parser in interpreting the actions. The set of inference rules constitutes a domain theory for DeJong's system, and includes knowledge about the goals and plans of human actors, as well as causal knowledge used to set up and verify expectations for future actions. The network of all the data dependency links created during the story parse is called an *inference justification network*, and corresponds to an explanation for the action sequence.

Generalization of the inference justification network is carried out by replacing general entities for the specific objects and events referenced in the network. As with the EBG method, the entities in the inference justification network are generalized as far as possible while maintaining the correctness of the data dependency links.[16] Then a new schema is constructed from the network. The issue of operationality enters into the process of determining an appropriate level of generalization for the schema constructed from the network. Should, for example, a generalized schema be created to describe the KIDNAPPING action sequence or the more general action sequences representing BARGAINING-FOR-MONEY or BARGAINING-FOR-WEALTH? All of these schemata explain the actions in the example story. DeJong cites several criteria to use in determining the level of generalization at which to represent the new schema (DeJong, 1983). The criteria include such considerations as: 1) Will the generalized schema be useful in processing stories in the future, or does the schema summarize an event that is unlikely to recur? 2) Are the preconditions for schema activation commonly achievable? 3) Will the new schema represent a more efficient method of achieving personal goals than existing schemata? Note that these schema generalization criteria roughly correspond to the operationalization criteria used in the EBG method. Because the generalized schemata are subsequently used for a story-understanding task, the operationality criteria pertain to that task.

[16] It is unclear whether the system regresses its equivalent of the goal concept through the inference justification network.

Table 5. The wealth acquisition schema generalization problem: Learning about ways to achieve wealth (DeJong, 1983)

Given:

- *Goal Concept:* The class of action sequences (i.e., a general schema) by which actor x can achieve wealth:

  WEALTH-ACQUISITION-ACTION-SEQUENCE (<a1, a2, ..., an>, x) ⇔
  NOT (WEALTHY (x, s0)) ∧ WEALTHY (x, EXECUTE (x, <a1, a2, ..., an>, s0))

  where

  <a1, a2, ..., an> is an action sequence; x is the actor; s0 is the actor's current state; and EXECUTE (a, b, c) returns the state resulting from the execution of action sequence b by actor a in state c.

- *Training Example:* The kidnapping story:
  FATHER-OF (FRED, MARY) ∧ WEALTHY (FRED)
  ∧ DESPERATE (JOHN) ∧ WEARS (MARY, BLUE-JEANS)
  ∧ APPROACHES (JOHN, MARY, s0) ∧ POINTS-GUN-AT (JOHN, MARY, s1)
  ...
  ∧ EXCHANGES (FRED, JOHN, $250000, MARY, s12)

- *Domain Theory:* Rules about human interaction, and knowledge about human goals, intentions, desires, etc.:
  FATHER-OF (person1, person2) → LOVES (person1, person 2)
  ∧ VALUES (person1, person2)
  WEALTHY (person) → HAS (person, $250000)
  EXCHANGES (person 1, person2, object1, object2) ⇔
  NOT (HAS (person1, object1)) ∧ VALUES (person1, object1)
  ∧ NOT (HAS (person2, object2)) ∧ VALUES (person2, object2)
  ∧ HAS (person1, object2) ∧ HAS (person2, object1)
  ...

- *Operationality Criterion:* Acquired generalization (i.e., the generalized schema) must satisfy the requirements for future usefulness in story-understanding (see text).

Determine:

- A generalization of training example (i.e., a generalized schema) that is a sufficient concept definition for the goal concept (i.e., that is a specialization of the wealth acquisition schema) and that satisfies the operationality criterion.

Table 5 summarizes the explanation-based generalization problem addressed by the explanatory schema acquisition research.

References

Borgida, A., Mitchell, T. & Williamson, K.E. (1985). Learning improved integrity constraints and schemas from exceptions in data and knowledge bases. In M.L. Brodie & J. Mylopoulos (Eds.), *On knowledge base management systems.* New York, NY: Springer Verlag.

DeJong, G. (1981). Generalizations based on explanations. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence* (pp. 67–69). Vancouver, B.C., Canada: Morgan Kaufmann.

DeJong, G. (1983). Acquiring schemata through understanding and generalizing plans. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence* (pp. 462–464). Karlsruhe, West Germany: Morgan Kaufmann.

DeJong, G. (1985). A brief overview of explanatory schema acquisition. *Proceedings of the Third International Machine Learning Workshop.* Skytop, PA, June.

DeJong, G., & Mooney, R. (in press). Explanation-based learning: An alternative view. *Machine learning.*

Dietterich, T.G., London, B., Clarkson, K., & Dromey, G. (1982). Learning and Inductive Inference. In P.R. Cohen, & E.A. Feigenbaum (Eds.), *The handbook of artificial intelligence.* Los Altos, CA: William Kaufmann, Inc.

Dijkstra, E.W. (1976). *A discipline of programming.* Englewood Cliffs, NJ: Prentice Hall.

Ellman, T. (1985). Explanation-based learning in logic circuit design. *Proceedings of the Third International Machine Learning Workshop.* Skytop, PA, June.

Fikes, R., Hart, P., & Nilsson, N.J. (1972). Learning and executing generalized robot plans. *Artificial intelligence,* 3, 251–288. Also in B.L. Webber & N.J. Nilsson (Eds.), *Readings in artificial intelligence.*

Kedar-Cabelli, S.T. (1984). *Analogy with purpose in legal reasoning from precedents.* (Technical Report LRP-TR-17). Laboratory for Computer Science Research, Rutgers University, New Brunswick, NJ.

Kedar-Cabelli, S.T. (1985). Purpose-directed analogy. *Proceedings of the Cognitive Science Society Conference.* Irvine, CA: Morgan Kaufmann.

Keller, R.M. (1983). Learning by re-expressing concepts for efficient recognition. *Proceedings of the National Conference on Artificial Intelligence* (pp 182–186). Washington, D.C.: Morgan Kaufmann.

Keller, R.M. (1985). Development of a framework for contextual concept learning. *Proceedings of the Third International Machine Learning Workshop.* Skytop, PA, June.

Keller, R.M. (1986). *Contextual learning: A performance-based model of concept acquisition.* Unpublished doctoral dissertation, Rutgers University.

Kibler, D., & Porter, B. (1985). A comparison of analytic and experimental goal regression for machine learning. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence.* Los Angeles, CA: Morgan Kaufmann.

Laird, J.E., Rosenbloom, P.S., Newell, A. (1984). Toward chunking as a general learning mechanism. (pp 188–192). *Proceedings of the National Conference on Artificial Intelligence.* Austin, TX: Morgan Kaufmann.

Laird, J.E., Rosenbloom, P.S., & Newell, A. (1986). SOAR: The architecture of a general learning mechanism. *Machine learning, 1,* 11 – 46.

Lebowitz, M. (1985). Concept learning in a rich input domain: Generalization-based memory. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach,* Vol. 2. Los Altos, CA: Morgan Kaufmann.

Mahadevan, S. (1985). Verification-based learning: A generalization strategy for inferring problem-decomposition methods. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence.* Los Angeles, CA: Morgan Kaufmann.

Michalski, R.S. (1983). A theory and methodology of inductive learning. *Artificial intelligence, 20,* 111–161. Also in R.S. Michalski, J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach.*

Minton, S. (1984). Constraint-based generalization: Learning game-playing plans from single examples. (pp 251–254). *Proceedings of the National Conference on Artificial Intelligence.* Austin, TX: Morgan Kaufmann.

Mitchell, T.M. (1978). *Version spaces: An approach to concept learning.* PhD thesis, Department of Electrical Engineering, Stanford University. Also Stanford CS reports STAN-CS-78-711, HPP-79-2.

Mitchell, T.M. (1980). *The need for biases in learning generalizations* (Technical Report CBM-TR-117). Rutgers University, New Brunswick, NJ.

Mitchell, T.M. (1982). Generalization as search. *Artificial intelligence*, March, *18*(2), 203–226.

Mitchell, T.M. (1983). Learning and Problem Solving. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence* (pp 1139 – 1151). Karlsruhe, West Germany: Morgan Kaufmann.

Mitchell, T.M. (1984). Toward combining empirical and analytic methods for learning heuristics. In A. Elithorn & R. Banerji (Eds.), *Human and artificial intelligence*. Amsterdam: North-Holland Publishing Co. Also Rutgers Computer Science Department Technical Report LCSR-TR-27, 1981.

Mitchell, T.M., Utgoff, P.E., & Banerji, R.B. (1983). Learning by experimentation: Acquiring and refining problem-solving heuristics. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning*. Palo Alto, CA: Tioga.

Mitchell, T.M., Mahadevan, S., & Steinberg, L. (1985). *LEAP*: A learning apprentice for VLSI design. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 573 – 580). Los Angeles, CA: Morgan Kaufmann.

Mooney, R. (1985). Generalizing explanations of narratives into schemata. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Mostow, D.J. (1981). *Mechanical transformation of task heuristics into operational procedures*. PhD thesis, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA.

Mostow, D.J. (1983). Machine transformation of advice into a heuristic search procedure. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine learning*. Palo Alto, CA: Tioga.

Nilsson, N.J. (1980). *Principles of artificial intelligence*. Palo Alto, CA: Tioga.

O'Rorke, P. (1984). Generalization for explanation-based schema acquisition. *Proceedings of the National Conference on Artificial Intelligence* (pp 260 – 263). Austin, TX: Morgan Kaufmann.

O'Rorke, P. (1985). Recent progress on the 'mathematician's apprentice' project. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Quinlan, J.R. (1985). The effect of noise on concept learning. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach*, Vol. 2. Los Altos, CA: Morgan Kaufmann. Modified, also in *Proceedings of the Second International Machine Learning Workshop*.

Rajamoney, S. (1985). Conceptual knowledge acquisition through directed experimentation. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Salzberg, S., & Atkinson, D.J. (1984). Learning by building causal explanations. *Proceedings of the Sixth European Conference on Artificial Intelligence* (pp 497 – 500). Pisa, Italy.

Schank, R.C. (1982). Looking at learning. *Proceedings of the Fifth European Conference on Artificial Intelligence* (pp 11 – 18). Paris, France.

Schooley, P. (1985). Learning state evaluation functions. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Segre, A.M. (1985). Explanation-based manipulator learning. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Shavlik, J.W. (1985). Learning classical physics. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Sims, M. (1985). An investigation of the nature of mathematical discovery. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Silver, B. (1983). Learning equation solving methods from worked examples. *Proceedings of the Second International Machine Learning Workshop* (pp. 99 – 104). Urbana, IL.

Soloway, E.M. (1978). *Learning = interpretation + generalization: A case study in knowledge-directed learning*. PhD thesis, Department of Computer and Information Science, University of Massachusetts, Amherst. Computer and Information Science Report COINS TR-78-13.

Tadepalli, P.V. (1985). Learning in intractable domains. *Proceedings of the Third International Machine Learning Workshop*. Skytop, PA.

Utgoff, P.E. (1983). Adjusting bias in concept learning. *Proceedings of the Eighth International Conference on Artificial Intelligence* (pp. 447 – 449). Karlsruhe, West Germany: Morgan Kaufmann.

Utgoff, P.E. (1984). *Shift of bias for inductive concept learning*. PhD thesis, Department of Computer Science, Rutgers University, New Brunswick, NJ.

Utgoff, P.E. (1985). Shift of bias for inductive concept learning. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach,* Vol. 2. Los Altos, CA: Morgan Kaufmann. Modified, also in *Proceedings of the Second International Machine Learning Workshop.*

Waldinger, R. (1977). Achieving several goals simultaneously. In E. Elcock & D. Michie, D. (Eds.), *Machine intelligence 8,* London: Ellis Horwood, Limited. Also in B.L. Webber & N.J. Nilsson (Eds.), *Readings in artificial intelligence.*

Watanabe, M. (1985). Learning implementation rules in operating-conditions depending on states in VLSI design. *Proceedings of the Third International Machine Learning Workshop.* Skytop, PA.

Williamson, K. (1985). Learning from exceptions to constraints in databases. *Proceedings of the Third International Machine Learning Workshop.* Skytop, PA.

Winston, P.H. (1985). Learning by augmenting rules and accumulating censors. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach,* Vol. 2. Los Altos, CA: Morgan Kaufmann.

Winston, P.H. Binford, T.O., Katz, B., & Lowry, M. (1983). Learning physical descriptions from functional definitions, examples, and precedents. *National Conference on Artificial Intelligence* (pp. 433–439). Washington, D.C.: Morgan Kaufmann.