Machine Learning 1: 145–176, 1986
© 1986 Kluwer Academic Publishers, Boston – Manufactured in The Netherlands

Explanation-Based Learning: An Alternative View

GERALD DEJONG (CSL!DEJONG@A.CS.UIUC.EDU)
RAYMOND MOONEY (CSL!MOONEY@A.CS.UIUC.EDU)
Coordinated Science Laboratory, University of Illinois, Urbana, IL 61801, U.S.A.

(Received December 30, 1985)

Key words: machine learning, concept acquisition, explanation-based learning

Abstract. In the last issue of this journal Mitchell, Keller, and Kedar-Cabelli presented a unifying framework for the explanation-based approach to machine learning. While it works well for a number of systems, the framework does not adequately capture certain aspects of the systems under development by the explanation-based learning group at Illinois. The primary inadequacies arise in the treatment of concept operationality, organization of knowledge into schemata, and learning from observation. This paper outlines six specific problems with the previously proposed framework and presents an alternative generalization method to perform explanation-based learning of new concepts.

1. Introduction

The explanation-based approach to learning has recently enjoyed increased attention in the literature (DeJong, 1983; Ellman, 1985; Minton, 1984; Mitchell, 1983; Mitchell *et al.*, 1985; Mooney & DeJong, 1985; Porter & Kibler, 1985; Shavlik, 1985; Silver, 1983). Its roots can be traced back to the MACROPS learning method in STRIPS (Fikes *et al.*, 1972), but its recent revival, extension, and refinement must be credited independently to Silver (1983), Mitchell (1983), and DeJong (1981). Each of these researchers concurrently developed rather large explanation-based learning computer systems: LEX2 (Mitchell), LP (Silver), and ESA (DeJong).

Mitchell, Keller, and Kedar-Cabelli (1986) present a lucid description of explanation-based generalization. Additionally, their formalization is shown to be able to encompass significant aspects of several other explanation-based systems, namely Winston *et al.*'s (1983) system which learns structural definitions from functional knowledge and the authors' system (DeJong, 1981; Mooney & DeJong, 1985) which learns schemata for natural language processing.

Our research on explanation-based learning has followed a somewhat different direction than that of Mitchell *et al.* Our approach is in some ways less general

and in some ways more general than that of Mitchell *et al.* (1986). The domains we have chosen have little in common with the domains of Mitchell's early work (Mitchell, 1983), so it is hardly surprising that we have reached different conclusions about explanation-based learning. As a result, we believe that Mitchell, Keller, and Kedar-Cabelli have omitted a number of important points concerning explanation-based learning.

We will argue that the following points should be taken into account by an explanation-based method.

(1) The broader term *Explanation-Based Learning* better describes the approach than does *Explanation-Based Generalization*. It seems both possible and desirable to apply the approach to concept refinement (i.e., specialization) as well as concept generalization.

(2) Mitchell *et al.*'s specification of explanation-based generalization does not specify how the examples' explanation is constructed. There are two possibilities: 1) the explanation can be constructed by an internal theorem prover/problem solver with no outside guidance or 2) the explanation can be constructed by observing and interpreting problem solving behavior of others. The two approaches require different levels of problem solving abilities, place different constraints on what concepts can be learned, and require different generalization steps.

(3) Mitchell *et al.*'s EBG generalization method is too weak. It often does not generalize the new concept far enough from the particular training example. The result is undergeneralizations that reflect many unimportant details of the example problem. As we shall see, this is a particular problem in several of the domains that we have been investigating.

(4) There are problems with the notion of an *Operationality Criterion* as advanced by Mitchell *et al.* The criterion itself is not always operational and should be dynamically computed rather than given as an input. We feel that the operationality criterion is only an approximation to an important, but much deeper, concept relating the abilities of the system's performance element to the learning process.

(5) Knowledge chunking or schematizing should play a central role in the generalization process. By asserting a hierarchical relationship among schemata, an explanation-based learning system can efficiently explore some alternative explanations of an example. This reasoning about schemata during the generalization process is not a part of the method advanced by Mitchell *et al.*

(6) Mitchell *et al.* have adopted a reduced version of goal regression to perform object generalization. They argue, quite correctly, that full goal regression (Waldinger, 1977; Nilsson, 1980) is inappropriate for explanation-based generalization. Instead of reduced goal regression, we advocate a different mechanism, equivalent in many ways to their reduced goal regression, but capable of unification retraction which allows further generalization and alteration of the explanation.

In our discussion we prefer the terminology of problem solving. Mitchell, Keller, and Kedar-Cabelli have cast their paper more in the terminology of theorem proving. These are two sides of the same coin. In the problem solving, one speaks of states of the world, the operators that transform one state to another, an initial state, the goal states which are in general partial characterizations of states, and the plans to achieve a goal from an initial state. In the theorem proving one speaks of the set of propositions believed, the rules of inference that transform one set of believed propositions to another (generally monotonically), the initial premises, a theorem to be proved, and the proofs constructed of a theorem. By identifying these constituents we can map any theorem proving problem into a problem solving task: given a theorem the equivalent "goal" is a state in which the theorem is among the propositions believed to be true. It is derived from the premises (the initial state) by applying rules of inference (operators) in a particular order. Thus, we will cast our comments in problem solving terminology with no loss of generality.

2. A brief overview of Mitchell *et al.*'s EBG method

Mitchell, Keller, and Kedar-Cabelli (1986) present a general technique for learning by generalizing explanations. The technique involves learning sufficient conditions for being an example of a particular concept by generalizing an explanation of why a particular example meets the definition of the concept. Explanations are represented as proof trees composed of horn-clause inference rules which conclude that the example is a member of the concept.

The generalization method they present (called EBG for Explanation-Based Generalization) must be provided with the following information:

(1) **Goal concept:** A definition of the concept to be learned in terms of high-level or functional properties which are not directly available in the representation of an example. For example, the following is a goal concept for a cup as given in Winston *et al.* (1983):

$$\text{OPEN-VESSEL}(x) \cap \text{STABLE}(x) \cap \text{LIFTABLE}(x) \supset \text{CUP}(x)$$

(2) **Training example:** A representation of a specific example of the concept in terms of lower level features. For example, a training example of a cup might include the following:

COLOR(OBJ1,RED)
LIGHT(OBJ1)
PART-OF(OBJ1,HANDLE1)

(3) **Domain theory:** A set of inference rules and facts sufficient for proving that a training example meets the high-level definition of the concept. The domain theory for the cup example might include the following inference rule:

$$\text{IS}(x,\text{LIGHT}) \cap \text{PART-OF}(x,y) \cap \text{ISA}(y,\text{HANDLE}) \supset \text{LIFTABLE}(x)$$

(4) **Operationality criterion:** A specification of how a definition of a concept must be represented so that the concept can be efficiently recognized. This assures that the learned definition of a concept is *operational* (Mostow, 1983) and can be readily used to recognize examples of the concept. The operationality condition for the cup example might be that the definition of an object be represented in terms of observable features of the object such as its weight and a specification of its physical parts.

Given this information, the first task in EBG is to construct an explanation of why the training example satisfies the goal concept by using the inference rules in the domain theory. This explanation takes the form of a proof tree composed of horn-clause inference rules which proves that the training example is a member of the concept. Since this explanation must have features of the original object description as its leaves, these leaves must satisfy the operationality condition. Next, this explanation is generalized to obtain a set of sufficient conditions for which this explanation structure holds in general. These conditions represent general operational conditions for being a member of the concept. The generalization process used in EBG is a specialization of *goal-regression* (Waldinger, 1977; Nilsson, 1980). By regressing the goal concept through the explanation structure, the desired general preconditions are obtained. The generalization process is not complete goal-regression as given by Waldinger since the structure of the original explanation is retained. Only a set of sufficient conditions, rather than necessary and sufficient conditions, are computed. Alternative ways of matching the antecedents of the rules and alternative ways of proving subgoals are not considered.

Mitchell *et al.* proceed to show how their EBG procedure can be used to generalize examples in different domains. Detailed explanations and generalizations are shown for three small examples. First is the SAFE-TO-STACK example in which a rule for when something can be safely stacked on a endtable is learned. Second is the CUP example (Winston *et al.*, 1983), in which a rule for recognizing a cup is learned. Third is a LEX2 example (Mitchell, 1983), in which a rule is learned for when it is useful to apply a certain integration operator. A summary of how EBG might be used to handle the authors' system which learns a schema for kidnapping (DeJong, 1981; Mooney & DeJong, 1985) is given in the appendix of Mitchell *et al.* (1986).

The final part of the paper gives a summary of future research issues for explanation-based generalization. These include the following:

(1) **Imperfect theory problem:** Methods are needed for building useful explanations in domains which have incomplete, intractable, or inconsistent theories.

(2) **Combining with similarity-based learning:** Techniques are needed which com-

bine explanation and similarity-based approaches to learning to achieve the benefits of both approaches.

(3) **Formulating generalization tasks:** Methods are needed for automatically generating generalization tasks by determining what needs to be learned to increase the performance of a system.

3. Explanation-based learning vs. explanation-based generalization

Mitchell *et al.* specify how the explanation-based approach can be used to generalize. Concept refinement is within the scope of the explanation-based approach as well. Mitchell *et al.* (1986) do not rule out explanation-based concept refinement, but neither do they discuss its possibility. We feel the term *Explanation-Based Learning*, which encompasses a broader range of phenomena, is to be preferred to *Explanation-Based Generalization* to describe the research.

Although we are not aware of any current systems which perform explanation-based refinement[1], it is an important research topic which we hope to address in the future. There are several reasons to pursue explanation-based refinement. The most important reason for us is that refinement can provide a partial solution to the difficult problem of generalizing non-independent conjunctive sub-goals.

Winston *et al.*'s (1983) cup program, discussed in Mitchell *et al.* (1986), represents a real world domain. In fact, the notion of a cup is very complex and messy. Winston's domain model, like all similar A1 models, necessarily only approximates the real world. It can be viewed as an abstraction space (Sacerdoti, 1974) of the real world. There are, therefore, examples where Winston's system disagrees with our real world concept of a cup. Consider a cup with a suitcase handle (much resembling a small pail – see Figure 1). This is clearly an acceptable kind of handle for Winston's system since in the original system the assertion of LIFTABLE is proved by resorting to previous knowledge of suitcases being liftable because of their handles. Yet this is clearly an overgeneralization since the pail-cup cannot easily be grasped and drunk from at the same time (Hofstadter, 1983). One might argue that this is not a failure in the proof of LIFTABLE but rather due to a poor initial functional specification of the cup concept. After all, it neglects the cup's most important function – enabling drinking. While this is perhaps a deficiency, it alone will not remove the pail-cup problem. After amending the functional specification of "cup" to include DRINKABLE-FROM, or some such predicate, a problem remains.

There is subtle and nasty interaction between the conjunctive sub-goals of LIFTABLE and DRINKABLE-FROM. The fact that a certain kind of handle makes a suitcase LIFTABLE without compromising the rest of its functionality does not

[1] A possible candidate is the OCCAM system of Pazzani (1985) which constructs a specialization for kidnapping infants after being given several examples in which infants are held for ransom.

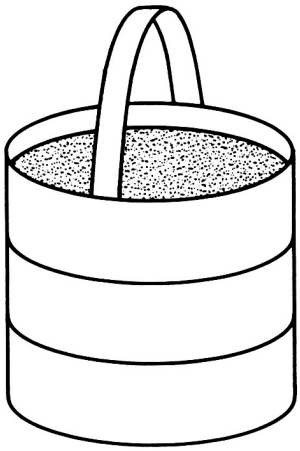

Figure 1. The pail-cup.

insure that it will do the same for cup. After all, the function of a suitcase is very different from that of a cup.

A very powerful inferencer would be able to keep track of all of the subtle implications among LIFTABLE, DRINKABLE-FROM, STABLE, and all other sub-goals. However, a powerful inferencer is not an appropriate solution to the "pail-cup" example. Such an inference engine would be exceedingly slow. It would spend vast amounts of time proving that many trivial and unlikely interactions do not occur and would take eons to process an example as complex as a drinking cup.

Explanation-based refinement provides a solution. One can use an inference engine that works in some abstraction space and is therefore quite efficient. In fact, Mitchell *et al.* (1986) propose constructing explanations using abstract theories to deal with the problem of intractable domains. However, abstract theories (like Winston's) will sometimes lead to over-generalization. One might choose, for example, to neglect conjunctive subgoal interactions. If one's planner never encounters contradictory examples, the simplifying assumptions are fine. On the other hand, if a failure is encountered when trying to use an incorrect concept, like the pail-cup, the system can inspect its previous proof to determine the source of the problem. In the case of the pail-cup, it will find that drinking is not possible because a hand is in the way. Furthermore, a hand is in the way because the location of the handle demands that location to satisfy LIFTABLE. Thus, the failure of an expected positive example can be used to guide the system through the original explanation proof, focusing attention at just those parts responsible for the offending interactions in the observed failure.

Thus, the explanation-based approach can be used to specialize over-generalized concepts. The example discussed illustrates how explanation-based specialization can aid in processing interacting sub-goals. However, we believe explanation-based

specialization will be useful in other areas as well. Since both specialization and generalization are subsumed, the broader term "Explanation-Based Learning" is more correctly used to describe the approach.

4. Explanation-based learning from observation

Learning from untutored observation is a very important type of learning (Carbonell, Michalski & Mitchell, 1983). It involves learning through monitoring of the world. A human "tutor" who presents and classifies examples for the system is not needed. Rather, the system is simply permitted to watch the behavior of others. It is the system's responsibility to decide which examples, and which features of those examples, are important.

Learning from observation is particularly promising for acquiring knowledge from human experts for knowledge-based systems. The expert is then unencumbered by the system which unobtrusively observes. The human needs no knowledge of the internal workings of the system, and the human expert is never asked to articulate the information or methods he uses – a step which has proven to be a bottleneck in current knowledge-based system development. Consequently, there has been an increasing amount of work in machine learning on *learning apprentice systems* (Minton, 1985; Mitchell *et al.*, 1985; O'Rorke, 1984; Wilkins *et al.*, 1985) which learn by observing expert problem solvers and generalizing the methods they use.

The generalization algorithm given by Mitchell *et al.* (1986) does not specify how the explanation is constructed. Yet to derive maximum benefit from the resulting generalization, the generalization algorithm should be sensitive to the explanation's derivation. As we shall see in section 9 the generalization algorithm is somewhat different for learning from observation than for learning from internal planning. The distinction consists in whether or not the explanation is known to be, in some sense, the "best" that can be supported by the domain theory. A system that learns by internal planning can be assured of generating an explanation that supports an optimal solution if, for example, it uses an admissible problem solving algorithm. If, on the other hand, the explanation is constructed by filling in the details of observed problem solving behavior of another, there is no guarantee that the resulting solution or its explanation are in any sense optimal.

To make this point more concrete consider a fanciful example. Suppose we know a fair amount about physics and electricity. Our domain theory consists of ohm's law, conservation of energy, and many other such concepts. It may have occurred to us that knowing how to convert electrical energy into light energy would be a useful thing. Thus, we have a general goal of knowing how to convert electricity to light and a domain model from which one can derive a number of explanations of how to achieve that goal. The system's task is to find and generalize one such explanation.

The first type of explanation-based learning system, which learns from internal

planning, requires a very powerful inferencer indeed. It is required to perform a massive search a task similar to the process of Edison's original invention of the lightbulb. The system might "invent" the neon or fluorescent light rather than the incandescent variety. An important point to note is that the efficiency of the solution (and the level of generality that the explanation will support) is determined entirely by the generality of the domain theory and properties of the planner. The LEX2 system (Mitchell, 1983) is of this variety. It requires no externally supplied suggestions as to how an integration problem is to be solved.

The second type of system forms explanations with the aid of external hints. In our electric light example we might give the hint "try tungsten." Another form of hint would be to specify all of the properties of a particular lightbulb: that its filament is tungsten, that its bulb is glass and that oxygen is evacuated from the bulb, etc. Incidentally, some of the example systems in Mitchell *et al.* (1986) (e.g., the SAFE-TO-STACK example) are of this type. However, they do not discuss the ramifications of this position either for the explanation construction process or the generalization process. A third type of hint might be the observed actions of an expert problem solver. To construct an explanation from an action sequence the system must justify that all preconditions are true (perhaps by tracing them to effects of previous actions). The system may also have to infer unobservable or unobserved actions and states. An example of this second type of explanation-based learning system is our natural language schema learning system (Mooney & DeJong, 1985). In an appendix, Mitchell *et al.* (1986) outline how this system can be cast into their framework. While the excercise clearly shows a deep grasp of our system, it is an unnatural fit. The *goal concept*, for example, is much more complex than the *goal concept* in the other systems mentioned. We believe the awkward fit is due to the lack of an observation input to their generalization method that can be used as an explanation hint.

Such hints can greatly reduce the amount of searching performed by the planner. Understanding complex problem-solving actions of someone else is a far easier task than producing those actions (DeJong, 1986). A system that learns from observation can possess a simpler planner and still acquire more difficult concepts than a system that must construct its own explanation from scratch.

Hints not only simplify the explanation process, they also bias the system toward certain explanations and away from others. For example, with the hint "try tungsten" the invention of the incandescent light, while still difficult, becomes much much easier. However, invention of the neon light becomes less likely. In the case of observing problem solving actions of an expert, the system is strongly biased towards an explanation of the particular solution used by the expert. The expert's solution may have obvious inefficiencies that will be reflected in the solution's explanation.

Note that in this case the efficiency of the solution (and also the level of generality that the explanation will support) is determined in part by the system's properties and in part by the hints it is given. It is important that the system's generalization process

factor out inefficiencies wherever possible. For example, we may observe an expert fashion the glass bulb for the incandescent light in a wasteful manner. If our domain theory includes a glass-blowing schema that specifies a better way to construct the bulb, the generalization process should substitute the efficient general schema for the inefficient observed action sequence. This step need not be performed when generalizing from an explanation constructed by a system of the first type: by constructing our planner and domain theory appropriately we can insure that the resulting explanations will have the desired properties.

There are two conclusions to be drawn from this section. First, the explanation-based generalization method should have an optional argument to allow for learning by untutored observation. This argument is a sequence of observed states and actions performed by an observed expert. The sequence need not be complete. It is treated as a hint at an underlying explanation for how to achieve the goal. Second, the generalization method should be somewhat different depending on whether the explanation was constructed from internal planning or observation.

5. Improving generalization

The generalization algorithm of Mitchell *et al.* seems to us to be very conservative. It seems to be primarily concerned with generalizing constants that appear explicitly in the training example into appropriately constrained variables. It appears from their description that predicates and constants which are introduced by inference rules are not subject to generalization. This view is born out by the examples in the paper.

The mechanism used in goal regression. However, full goal regression is an unwieldy process and is not performed. Instead, EBG uses a modified version of goal regression which ignores disjuncts. If there are several alternative variable identifications that allow a proof to be true, only the unification used in the explanation of the training example is considered. Also, if there are several possible ways to complete a subproof (which would result in regression through disjunctive subgoals) only the subproof actually used in the training example's explanation is used.

The result of these design decisions is that the resulting concept description is not as general as it might be. Indeed, specific and unimportant details of the example observation can be reflected in the generalized concept.

As we shall see in section 8 many of these problems can be overcome through integrating the performance element into the generalization algorithm. It is instructive to discuss two important types of generalization that goal regression alone cannot perform.

5.1. Generalizing predicates

The first problem we will discuss is that of generalizing predicates. This problem does not arise easily in the "SAFE-TO-STACK" example or other similar domains. However, it arises immediately in moderately complex problem-solving domains. Consider our natural language system (Mooney & DeJong, 1985) which is given the following kidnapping example and must learn a general kidnapping concept:

Fred is the father of Mary and is a millionaire. John approached Mary. She was wearing blue jeans. John pointed a gun at her and told her he wanted her to get into his car. He drove her to his hotel and locked her in his room. John called Fred and told him John was holding Mary captive. John told Fred if Fred gave him $250,000 at Trenos then John would release Mary. Fred gave him the money and John released Mary.

The story includes a description of the kidnapper telephoning the victim's father. The telephone event might be represented as

TELEPHONE(human079, human077, cond003)

where human079 is John, the kidnapper, human077 is Fred, the millionaire, and cond003 represents the bargain conditions that John proposes.

An inference rule of the domain theory might be

TELEPHONE(x,y,z) $\supset$ KNOWS(y,z)

That is, successfully telephoning someone results in a particular mental state of the listener. The knowledge state is crucial to the success of the kidnapping. Thus, the TELEPHONE action and resulting knowledge state are components of the explanation. The millionaire must believe that giving money will free the victim. Otherwise the action of John giving Fred the money is unmotivated.

EBG's goal regression method allows no possibility of altering the predicates that compose the explanation. Using this generalization method on the example results in the kidnapping concept which *requires* the use of a telephone to inform the millionaire. This is clearly an extreme undergeneralization. There is nothing central about telephoning. The important part of the explanation is the knowledge state that the kidnapper achieves in the millionaire. Telephoning is just one of many ways to achieve that state. He might as well have sent a ransom note, or a telegram, or a carrier pigeon. Indeed, any communication mechanism that achieves the required knowledge state would work. The concept of kidnapping ought to reflect this generalization. The TELEPHONE action predicate of the observed example ought to be replaced by a more general COMMUNICATE action predicate in the general concept of kidnapping. The COMMUNICATE action can then be instantiated at

problem-solving time in one of many possible ways depending on the state of the world at the time the problem-solver invokes KIDNAPPING. If a phone is handy, it can be used; otherwise another method can be selected.

There are other eccentricities of the example that the generalization algorithm of Mitchell *et al.* would preserve: only daughters of millionaires can be kidnapped, the ransom payment must take place at a restaurant, etc. These inappropriate dependencies on the details of the observed example must be eliminated by the generalization process if the system is to have any chance of applying its new knowledge productively. These details are not entirely irrelevant; their effects are crucial to the success of the observed concept. However, requiring the precise details that were observed seems to narrow the concept unnecessarily.

5.2. Structural generalization

A related but separate issue involves the structure of the example explanation. The EBG method does not permit generalization of the example explanation's structure. Consider the "SAFE-TO-STACK" concept. In the observed example SAFE-TO-STACK is true because the first object is proved to weigh less through a calculation based on its volume and density. This is fine for this specific example. The problem is that the resulting general concept "SAFE-TO-STACK" then *requires* the first object's weight to be calculated by a volume times density computation. Suppose our world knowledge specifies that the default weight of an ashtray is 1 (in the same units as the default weight of an endtable), and that OBJ17 is an ashtray whose density is unknown. The SAFE-TO-STACK concept acquired by the EBG method cannot itself conclude that the ashtray can safely be stacked on the endtable.

The complete explanation must be present. No alternative achievement of subgoals is allowed, no matter how easy it may be to construct such alternatives. This inability to alter the example's explanation structure results in another kind of undergeneralization. While related to the previous undergeneralization problem it cannot be alleviated by a one-for-one substitution of a more general schema for a more specific one.

To illustrate the inefficiency consider a robot manipulator system (Segre & DeJong, 1985) that learns assembly concepts (i.e., how to assemble parts from an explanation-based generalization of an observed example).

First it is necessary to develop some background on our manipulator model. There are two low-level manipulator actions to change the position of a grasped piece: TRANSLATE and MOVETO. TRANSLATE moves the grasped piece along a straight line while maintaining the piece's orientation. It is an expensive and slow operation since the kinematic equations must be solved at many points with interpolations in between to insure the proper orientation and straight line trajectory. MOVETO changes the current position and orientation of the grasped piece to some

desired position and orientation. MOVETO is faster and more efficient. It solves the kinematic equations only once for the desired state and then sets all of the manipulators joints accordingly. It is cheaper to use than TRANSLATE because it does not enforce any specific intermediate values on the position or orientation of the grasped piece.

Now we can examine how inefficiencies can arise in a task assembly concept acquired by an explanation-based method similar to that of Mitchell *et al.* (1986). Suppose our learning manipulator is shown how to build a miniature bench by placing piece A in just the right spot on the top of piece B so that the pegs in A are inserted into the holes in B (see Figure 2). Further, suppose that the assembly sequence demonstrated to the system consists of grasping piece A, followed by a TRANSLATE to lift piece A straight up off of the table, a TRANSLATE to position piece A directly above piece B, a ROTATE to orient piece A to correctly match piece B, and finally a TRANSLATE down to place piece A on piece B (see Figure 3A).

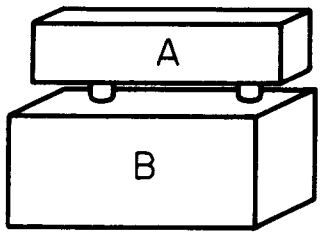

Figure 2. The miniature bench.

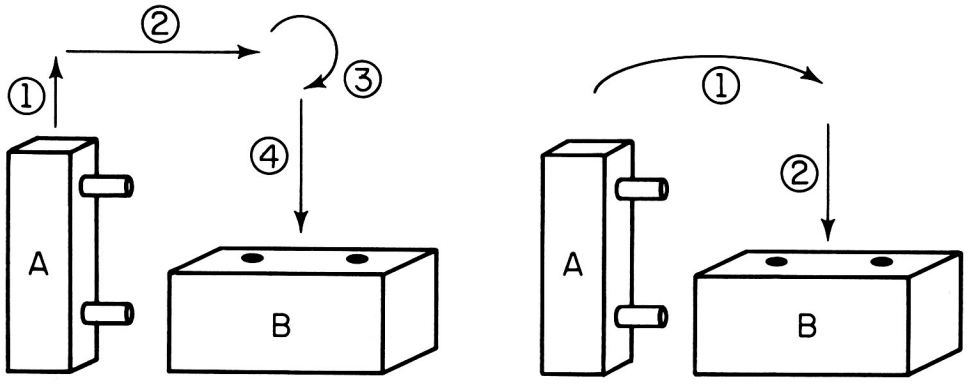

Figure 3A. Inefficient sequence.
Figure 3B. Efficient sequence.

The explanation of how the bench is realized by the observed action sequence includes, of course, the TRANSLATE and ROTATE actions. The explanation specifies that the last TRANSLATE combines the pieces while maintaining their proper relative orientations: the ROTATE establishes the proper orientation of piece A with respect to piece B; etc. Consider using the EBG procedure to generalize this explanation. Since all of the actions participate in the explanation or proof that a correct bench is constructed, they all appear in the new BUILD-A-BENCH task concept generated by the EBG procedure. Generalization has occurred so that the piece identity with A and B has been eliminated. However, since the EBG process cannot alter the structure of the explanation, the generalized concept achieves the bench using three TRANSLATEs and a ROTATE, even though a more efficient MOVETO could be used instead of the first two TRANSLATEs and the ROTATE. The MOVETO equally well positions piece A above piece B with the proper orientation (see Figure 3B). An efficient task concept is learned because the observed task sequence is inefficient. While it is unrealistic always to expect an optimal task concept from observing a horribly inefficient example, we will see in section 8.1 that some optimizing is easily done. In particular, in this case the system's domain theory must already contain the knowledge necessary to replace the TRANSLATEs and ROTATE with a MOVETO. This knowledge has to do with the effects of these actions and must already be present to perform the goal regression. Yet the knowledge cannot be used by the EBG procedure to fabricate a more efficient assembly concept.

6. Problems with the operationality criterion

How can we assure that a new concept is meaningful to the problem solver? The final operational definition of a concept must be specified in a vocabulary which the problem solver understands. After all, if recognizing or applying a new concept involves solving sub-problems which are beyond the capabilities of the problem solver, then that concept will not improve the efficiency or abilities of the problem solver.

Mitchell *et al.* (1986) recognize this as an important problem. Their solution is to restrict the allowable target vocabulary through the *Operationality Criterion*. This specifies that "The concept definition must be expressed in terms of the predicates used to describe examples ... or other selected, easily evaluated predicates from the domain theory ..."

There are two problems with this definition of the operationality criterion. First, the process of specifying the operationality criterion for a particular domain is not itself operational. It is clear that Mitchell *et al.* (1986) have in mind including some predicates and not others and that the problem solver should have more or less immediate access to the truth values of the predicates which are included. However, there is no mention of how these predicates are actually selected or how easy their evaluation must be to meet the operationality criterion.

Second, the specification of predicates alone will not insure ease of evaluation. The difficulty of discovering a truth value is properly associated with a proposition as a whole, not just the predicate. Consider, for example, the following two propositions:

(PROVABLE"2+2=4")
(PROVABLE"FERMAT'S LAST THEOREM")

They both use the same predicate but the truth value of the first is far easier to determine than the truth value of the second. The first is operational, the second is not. Many predicates behave in this fashion; their operationality depends on their arguments. A training example might be selected in which the propositions are easily evaluated but the very same predicates may be difficult to evaluate when applied to later examples. Nor can we limit the operationality criterion to predicates that are known to be operational for all arguments. Such a restriction would overly impoverish the vocabulary. Predicates like VOLUME and DENSITY necessary for the SAFE-TO-STACK example could not be used. Determining the volume of an irregularly shaped solid can be quite involved. It is easily evaluated only for simple objects or objects whose volume is included as innate in the domain theory or explicitly specified in the training example.

7. A technical problem with the EBG algorithm

In addition to the conceptual difficulties we have raised, there is a technical problem with the generalization algorithm as described in Mitchell *et al.* (1986). The problem is that they fail to mention the necessity of an additional step when generalizing an explanation. The following simple example illustrates the problem. The rules in the domain theory for this example are:

$$\text{HATE}(a,b) \cap \text{POSSESS}(a,c) \cap \text{WEAPON}(c) \supset \text{KILL}(a,b)$$
$$\text{DEPRESSED}(w) \supset \text{HATE}(w,w)$$
$$\text{BUY}(u,v) \supset \text{POSSESS}(u,v)$$
$$\text{GUN}(z) \supset \text{WEAPON}(z)$$

Given the following information (i.e. training example):

DEPRESSED(JOHN)
BUY(JOHN,OBJ1)
GUN(OBJ1)

it is easy to conclude that John will commit suicide as shown in Figure 4. The result of performing EBG generalization on this proof is shown in Figure 5 in the manner

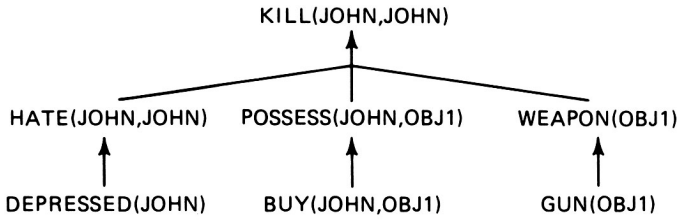

Figure 4. Explanation of the suicide example.

used in Mitchell *et al.* (1986). Each conjunct in the current regressed expression (the underlined elements at each level) is unified with the consequent of the appropriate rule to yield a set of substitutions. This set of substitutions is then applied to the antecedent of the rule to produce the new regressed expression. The substitution is also applied to the remaining conjuncts before regressing them through their appropriate rules. The result is the following set of generalized conditions:

$$\text{DEPRESSED}(y) \cap \text{BUY}(y,c) \cap \text{GUN}(c)$$

However, the goal concept specified in the explanation structure is KILL(x,y) when in fact the explanation is only valid for the specialized concept of suicide: KILL(y,y). The problem is that regression only results in a set of generalized antecedents, or sufficient conditions for being an example of the concept. It does not result in a specification of the proper goal concept. As in the SUICIDE example, the explanation itself may impose certain constraints on the goal concept. Although the problem of determining the appropriate generalized goal concept is not discussed at

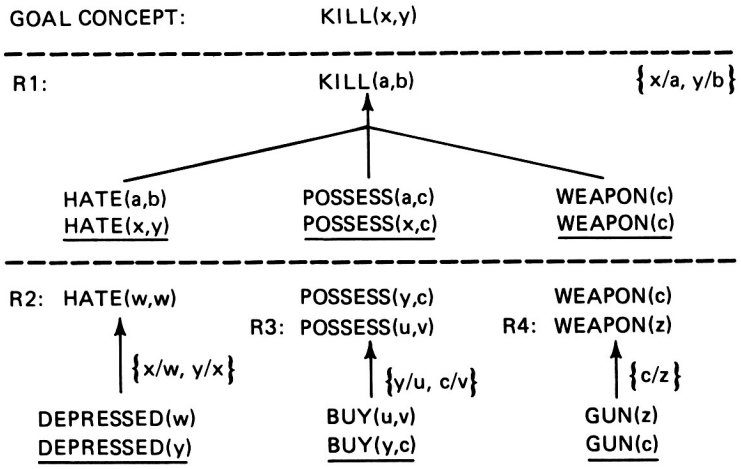

Figure 5. Generalizing the suicide example using EBG. (Underlined expressions are the results of regressing the goal concept.)

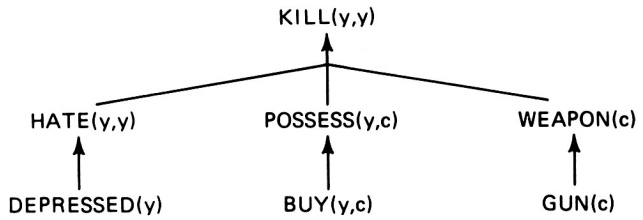

Figure 6. The generalized proof for the suicide example.

all in Mitchell *et al.* (1986) it is discussed in Mahadevan (1985), and Mitchell *et al.* (1985). In these papers, they described how to determine the generalized goal concept by rederiving a generalized proof, using the regressed antecedents as a training example and applying the same rules as in the original example. The regenerated proof for the suicide example is shown in Figure 6. The complete generalization algorithm using goal regression is actually a two step process:

(1) Regress the original specification of the goal concept through the explanation to obtain a set of generalized antecedents.
(2) Starting with these generalized antecedents, rederive the proof to obtain the generalized goal concept (consequent).

In a later section we will present an alternative generalization algorithm which does not employ goal regression and avoids having to make two passes through the explanation.

8. Some solutions

The solution to some of these dilemmas involves acknowledging that the explanation-based learning system has two logically distinct components: an acquisition element and a performance element. The purpose of learning is to improve the processing ability of the performance element. The acquisition element includes two phases. First, it constructs an explanation for the training example and then it generalizes the explanation to form a new concept. The performance element makes use of the new generalized concept. In learning from observation, the explanation phase of the acquisition element produces an explanation from the observed operator sequence. In learning from one's own plans, the explanation phase of the acquisition element uses the performance element. Either way, the generalization phase of the acquisition element should be sensitive to the abilities of the performance element. The notion of operationality must be judged with respect to the abilities of a particular performance element. This is not the case in Mitchell *et al.* (1986). While both the understanding and the performance element operate on the same knowledge base, they reflect the world knowledge in subtly different ways.

The understander's task is to construct a causal model of the observed example. The explanation of how the example satisfies the goal in question is a subset of the causal model. The performance element benefits from the generalized explanation. As more concepts are acquired the performance element is able to process much more efficiently and, due to resource limitations, is able to correctly process observations that were previously beyond its capabilities. The performance element can be viewed as a problem solver in the broad sense of the term. It achieves goals by finding appropriate action sequences (as in the robot manipulator system) or recognizes instances of concepts (as in the SAFE-TO-STACK example). In either case the performance element has a task with a well defined goal and performance on this task is to be improved through explanation-based learning. We will find it useful to refer to the performance element as a problem solver and to use the vocabulary of problem solving to describe its operations.

Since the functioning of the problem solver (or performance element) is important only after the concept has been learned, Mitchell *et al.* leave it out of their specification of the Explanation-Based Learning process. However, we believe the abilities of the problem solver should have a major impact on the generalization process and therefore on the concept to be learned.

8.1. Generalizing with schemata

In an explanation-based system which learns schemata, the schema knowledge itself can be used to generalize the example. This type of generalization is not a part of the explanation-based generalization method outlined in Mitchell *et al.* (1986), but as we shall see, it can be employed to overcome some of the undergeneralization problems discussed in the previous section. This form of generalization is possible because the schemata are themselves generalized knowledge chunks.

The central idea underlying a schema-based problem solving system is that the system's knowledge should be structured in such a way that knowledge relevant to achieving a certain goal is grouped together (Chafe, 1975; Charniak, 1976; Minsky, 1975; Schank & Abelson, 1977).

This structuring of the system's knowledge into schemata aids in understanding observed inputs and also results in an important, and efficient kind of generalization. For example, in the kidnapping story given earlier, John (the kidnapper) points a gun at Mary (the victim) to force her into his car so he can drive her to his hotel where he detains her. This is an important action for John; if he did not have the gun Mary would probably not have gotten into the car and his attempt at kidnapping would have failed. However, this action is not essential in all kidnappings. There are many ways John might have forced Mary into his car. He might have threatened her with a knife or some other weapon, he might have overpowered her, or he might have drugged her. Furthermore, he might not have used his car at all; he might have simply

tricked her into entering his hotel room to capture her, etc. Clearly a general kidnapping schema ought not to require a gun. The gun-pointing event should be generalized in the final schema.

Is there an appropriate generalization of the gun-pointing event? Can it be generalized efficiently without tediously replanning with all possible variations? The answer to both questions is "yes", and the trick is to rely on the schematic structure of the example's explanation.

In a schema understander the explanation is constructed by finding how the observed inputs fit into known schemata and determining how these schemata can fit together into a well-formed whole. The pointing of the gun must be understood as a component of the known schema THREATEN. Furthermore, it must be understood that John is THREATENing Mary so that she will get into his car, and he wants her in his car so he can drive her to his hotel room to lock her up. Thus, pointing a gun is John's sub-sub-sub-sub-goal of detaining Mary. In our system, the schema for seizing and detaining someone is called CAPTURE. In constructing the explanation for this story the system realizes that a CAPTURE of Mary must be performed to allow a plausible BARGAIN between John and Fred in which John trades Mary's freedom for $250,000. Thus, the system understands that John uses the known schema CAPTURE in a novel way to satisfy a precondition for the known schema BARGAIN.

The schemata CAPTURE and BARGAIN are already known and, therefore, already generalized. This already-existing generality can be exploited in the new schema for kidnapping. The solution is to eliminate those portions of the explanation which are merely nominal instantiations of known schemata. In the kidnapping explanation the system is left with CAPTURE and BARGAIN. The goal regression step, or its equivalent, is then performed on this greatly reduced explanation.

What happened to the event of John pointing his gun at Mary? It was eliminated as a nominal (and, therefore, easily replannable) sub-goal deep inside the explanation of the CAPTURE schema. However, any other successful instantiation of CAPTURE would have worked as well, provided it satisfies the inter-schema requirements imposed by the goal regression phase (i.e., that there must be a rich person who values the freedom of the person selected to be captured, etc.)

8.2. A better operationality criterion

Of course, the CAPTURE schema itself need not contain a complete specification of all possible ways to CAPTURE a person. Rather it need only provide CAPTURE specific information which can then be easily fleshed out in a number of ways by the performance element through interactions with the system's other schemata (Chafe, 1976; Schank, 1982).

Any goal for which the system possesses a schema can be operationalized with very

little effort. Thus, the operationality criterion is exactly that set of goals for which the system possesses schemata. In other words, the operationality criterion is that set of goals that the system could easily achieve using normal means by simply instantiating an existing schema. There are two important implications of this position: 1) The operationality criterion is derivable from the system's performance element and, therefore, should not be independently input to the system; 2) The operationality criterion is dynamic, not static; as the system learns new schemata, additional goals become operational since the system can use the new schemata as building blocks to construct future explanations.

8.3. An alternative to goal regression

As described earlier, Mitchell *et al.*'s EBG method employs a version of goal regression to generalize explanations. As we explained in section 7, the generalization process described in Mitchell *et al.* (1986) has a technical problem which can be corrected by augmenting the process. In this section, we present an alternative technique for generalizing explanations which we believe has certain advantages.

We developed our method independently of Mitchell *et al.* and it does not employ goal regression. The technique involves maintaining a version of the proof tree with uninstantiated versions of the general rules (an *explanation structure* (Mitchell *et al.*, 1986)) and two independent substitution lists called SPECIFIC and GENERAL. Applying the SPECIFIC substitution to the *explanation structure* results in an explanation for the specific example, while applying the GENERAL substitution results in a generalized explanation.

The explanation structure and the two substitutions lists are maintained during the construction of the original explanation independently of how the explanation is built. When a rule is invoked during the explanation phase, whether by a planner or by an understander, a copy of the uninstantiated rule is added to the explanation structure. A unification between a specific statement or goal of the particular example and a general domain rule is made in the context of the SPECIFIC substitution and any new substitutions required for the unification are added to the SPECIFIC list. A unification between two domain rules in the explanation structure is done twice, once using the SPECIFIC substitution list and once using the GENERAL one, in both cases adding any newly created substitutions. This process assures that the GENERAL list contains only and all those substitutions which are required to maintain the validity of the explanation structure.

Letting σ stand for the SPECIFIC substitution and γ for the GENERAL substitution, the generalization procedure may be formally specified as follows (using the notation defined in Nilsson (1980)):

for each equality between expressions e_1 and e_2 in the explanation structure:
  if e_1 is the antecedent of a domain rule **and** e_2 is the consequent of a domain rule
  then let ϕ = the most general unifier of $e_1\sigma$ and $e_2\sigma$
    let $\sigma = \sigma\phi$ (* update SPECIFIC substitution)
    let δ = the most general unifier of $e_1\gamma$ and $e_2\gamma$
    let $\gamma = \gamma\delta$ (* update GENERAL substitution)
  else let ϕ = the most general unifier of $e_1\sigma$ and $e_2\sigma$
    let $\sigma = \sigma\phi$ (* update SPECIFIC substitution)

We have constructed a prototype generalizer which uses this procedure to generalize explanations in various domains (Mooney & Bennett, 1986). This system has been used to generalize examples from the literature on STRIPS robot planning (Fikes *et al.*, 1972), generating physical descriptions of objects from functional information (Winston *et al.*, 1983), solving integration problems (Mitchell, 1983), designing logic circuits (Mitchell *et al.*, 1985), and proving theorems in mathematical logic (O'Rorke, 1984). Figure 7 shows our generalization process applied to the suicide example. The final substitutions shown in the figure are obtained by performing the unifications for rules R1 through R4 in order, although any order results in equivalent substitutions up to a change of variable names. Table 1 shows a trace of the unifications and how they affect the two substitutions. Applying the SPECIFIC substitutions list to the explanation stucture generates the specific explanation as shown in Figure 4. Applying the GENERAL substitution list to the explanation structure generates the generalized explanation (the same as Figure 6 but with y renamed to w and c renamed to z).

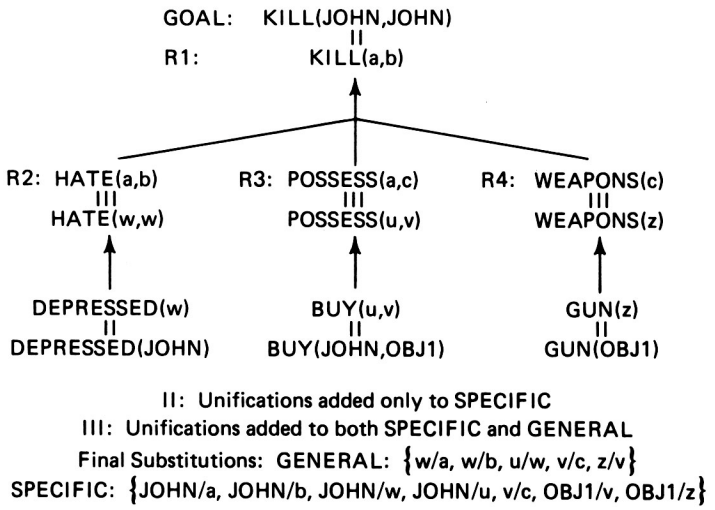

Figure 7. Explanation structure and substitutions for the suicide example.

Table 1. Unifications for the suicide example.

Unification	Specific	General
R1:		
KILL(JOHN,JOHN) = KILL(a,b)	JOHN/a,JOHN/b	
R2:		
HATE(a,b) = HATE(w,w)	JOHN/w	w/a, w/b
DEPRESSED(w) = DEPRESSED(JOHN)		
R3:		
POSSESS(a,c) = POSSESS(u,v)	JOHN/u, v/c	u/w, v/c
BUY(u,v) = BUY(JOHN,OBJ1)	OBJ1/v	
R4:		
WEAPON(c) = WEAPON(z)	OBJ1/z	z/v
GUN(z) = GUN(OBJ1)		

The general rule represented by this generalized proof is:

$$\text{DEPRESSED}(w) \cap \text{BUY}(w,z) \cap \text{GUN}(z) \supset \text{KILL}(w,w)$$

The technique outlined above is functionally equivalent to the corrected goal-regression technique described in section 7. In other words, the generalized antecedents generated from a given explanation are the same for both algorithms. However, we believe there are several advantages of this technique over the goal-regression approach. First, the generalization process is integrated with the process for building explanations, instead of requiring an independent process which must trace down the explanation after it is constructed. Every time a rule is added to the explanation, the SPECIFIC and GENERAL substitutions are updated. Generalization is accomplished by simply applying a substitution to the explanation structure. Second, it results in a full generalized explanation. Goal regression, on the other hand, only results in generalized antecedents. In order to obtain the full generalized explanation using goal regression, one must construct it by reapplying the rules in the explanation structure to the generalized antecedents (as in Mahadevan (1985)). It is not clear whether or not our method is actually more efficient than EBG in terms of computational complexity; however, we feel these two differences at least make our approach more elegant.

As additional facility of our generalization method is the ability to retract unifications from the GENERAL substitution. This is important in that it allows for "pruning" of the explanation if the "operationality criterion" is met by higher nodes in the proof tree and not just the leaves. For example, if we knew that we could easily determine whether or not a given item was a WEAPON, we could further generalize the suicide example by not requiring that the weapon be a GUN. This is done by removing the inheritance rule: $\text{GUN}(z) \subseteq \text{WEAPON}(z)$ from the generalized explanation and retracting its unifications from the GENERAL substitution. This

results in retracting the substitution: z/v from the GENERAL substitution shown in Figure 7. The new generalized rule for this example then becomes:

$$\text{DEPRESSED}(w) \cap \text{BUY}(w,v) \cap \text{WEAPON}(v) \supset \text{KILL}(w,w)$$

When an explanation is constructed by observing external behavior, it is important to be able to "prune" the explanation since the system may already have operational concepts which are more general than those used in the particular training example.

9. Our version of explanation-based generalization

We now advance a modification of the explanation-based generalization problem and method given in Mitchell *et al.* (1986). We will assume that the performance element is a schema-based problem solving system. Within this context, our method attempts to overcome some of the problems outlined earlier in the paper. In particular, it includes the possibility of learning from observation, it allows for generalizing predicates, and it allows for altering the structure of the explanation. It also eliminates the *operationality criterion* as an explicit input.

First, we will require some additional terminology. The world is composed of *objects*, each of which may have properties and relations to other objects. A specification of all of the objects and their current properties and relations is a *world state.* A *goal* is a specification of a desired partial world state. *Operators* map one world state into another by altering the properties or relations of objects in the world. An operator is defined by specifying its effects and the conditions which must hold in the world for its application. It is assumed that operators must be applied serially, not in parallel. A *schema* is a partially ordered set of operators and/or simpler schemata linked causally. For the most part, schemata will be used as problem solving concepts. That is, useful schemata will be those that organize operators to achieve an important goal, or set of goals, in a general way. A *problem* is specified by an initial state and a goal. The *solution* to a problem is a sequence of operators which transform the initial problem state into some state which matches the goal. An *explanation* of a solution is the operator sequence that solves a problem, together with an annotation which captures how effects of one operator match the preconditions of another. Often this annotation takes the form of matching the expansion of a known schema to operators or lower-level schemata. When no known higher-level schema can incorporate a lower-level operator/schema sequence, the annotation is simply a specification of the most general unifications necessary for the effects of antecedents actions to match the preconditions of later ones. The explanation forms a proof that the solution sequence does, in fact, achieve the goal.

Table 2 presents a summary of the generalization problem we are interested in. The task faced by the system is to augment its domain theory with a new schema which

specifies a general method for achieving the goal. This is done by finding a particular solution to the given problem and then generalizing the solution into a schema using the system's existing domain theory. The particular solution may either be input to the system or generated by the learning system's performance element. If the solution is produced internally then the *observed operator sequence* is not given.

Table 2. The explanation-based generalization problem.

Given:

- *Domain Theory:* The domain theory consists of three parts. First, a specification of types of objects in the world and their properties. Second, a set of inference rules for inferring properties and relations from other properties and relations. Third, a library of problem solving operators and already known general schemata. These schemata can be previously learned or hand-coded.
- *Goal:* A general specification of a goal state. In general, a goal is an incomplete world state and can be specified in non-operational terms (e.g., in a functional vocabulary).
- *Initial World State:* A specification of the objects in the world and their properties.
- *Observed Operator/State Sequence* (optional): An observed sequence of low-level operators, performed by an expert, which achieves an instance of the goal. In some situations, some operators may be missing, in which case they must be inferred from achieved states given in the input.

Determine:

- A new schema that achieves the goal in a general way.

Table 3. The generalization method.

(1) If there is no observed operator input, use the domain theory to construct a plan which achieves the goal. Otherwise, use the domain theory to build a causally complete interpretation of the observed operator input. In either case, maintain GENERAL and SPECIFIC substitutions as described in section 8.3.

(2) Eliminate operators and states which do not causally support the goal. The remaining structure is the solution explanation.

(3) Identify nominal instantiations of known schemata and eliminate them by retracting the unifications between the general specification of the schema and the specific operators used in the instantiation.

(4) Remove actions and states which only support inferences to more abstract actions or states. This is done by removing these abstraction inferences along with their antecedents and retracting the unifications binding them to the rest of the explanation.

(5) If the explanation was constructed from an observed solution, look for subgoals which can be achieved in a more efficient manner. For each subgoal in the explanation, check if the system already has a schema for achieving that subgoal which is more efficient than the one used by the observed agent. If so, retract the observed plan for achieving this subgoal and insert the more efficient schema.

(6) Generate the final generalized explanation by applying the resulting GENERAL substitution to the remaining explanation structure.

9.1. The explanation-based generalization method

Table 3 is an outline of our explanation-based generalization method. Steps 1 and 2 form the explanation phase. If an expert is observed solving the problem, the method builds an interpretation of the input operators. It justifies how the goal is realized and how the preconditions of each operator are achieved. If there is no observed operator input, the system constructs its own plan to achieve the goal. Step 2 eliminates observed operators and initial states which are irrelevant to the realization of the goal. For example, if an apprentice system observes an expert air traffic controller buy a cup of coffee from a nearby vending machine, it disregards that action as not supporting the goal of avoiding mid-air collisions. If a block's world planning system constructs a plan for building an arch, it eliminates facts about the color of the blocks since these facts are never used as preconditions of an action in the plan.

Step 3 insures that no redundant work is done in generalizing an example. It eliminates nominal instantiations of already known schemata. A nominal schema instantiation is one in which the instantiation gives no significant information about what happened beyond that already specified in the general schema. For example, consider John as an expert problem-solver who, in the course of his problem solving, must eat lunch. Suppose we, as a learning apprentice, observe him entering McDonalds on Green Street. He walks up to the counter, asks the attendant for a cheeseburger, reaches into his pocket for money, pays for the cheeseburger, sits down at a table, eats, etc. In short, he does nothing out of the ordinary. All of his actions fit neatly into our already existing McDonalds schema. His actions can be easily rederived from the general McDonalds schema together with its variable bindings of "Green Street" and "cheeseburger" because all McDonalds are alike. There is no point in re-discovering a general version of the McDonalds schema from the low-level observed actions of John. That work would be wasted; we already have a general McDonalds schema. In general, for any subgoal in the explanation which is achieved by specifying a known schema and how that schema was instantiated in a nominal way, the actions making the nominal instantiation can be eliminated. The only structure that needs to be kept is the identity of the general schema and its variable bindings.

Step 4 further generalizes the explanation by eliminating all but the most abstract actions and states which support the achievement of the goal. This step eliminates abstraction inferences like the GUN $\supset$ WEAPON inference which was eliminated from the SUICIDE example in section 8.3. It also provides an effective way of generalizing the actions in the explanation, like the TELEPHONE to COMMUNICATE generalization discussed earlier. Since TELEPHONE only supports the eventual goal through a BELIEF state which is inherited from the more abstract action COMMUNICATE, the TELEPHONE may be eliminated from the explanation leaving only the more abstract action COMMUNICATE.

If the explanation is generated from observing external problem solving, step 5 is

an attempt to improve the efficiency of a possibly sub-optimal plan. This is done by searching for subgoals which the system already has efficient schemata for achieving. If the system has a more efficient method for achieving a particular subgoal, it is substituted for the set of operators used by the observed agent. This allows for the substitution of a MOVETO for two TRANSLATEs and a ROTATE in the robot assembly task discussed in section 5.2.

Step 6 simply applies the GENERAL substitution to the remaining explanation structure to give the final generalized explanation. The resulting generalized explanation can be easily transformed into a new schema by identifying its preconditions and effects. The preconditions of the new schema are those preconditions of the component schemata and operators which are not guaranteed to be satisfied internally. Some of its effects are simply the effects of the component schemata and operators not eliminated by a later component. However, some of the effects are specializations of the component's effects. These constraints are due to the required unifications with other components imposed by the causal structure of the solution.

9.2. The kidnapping example

As an example of the above procedure, consider the GENESIS natural language system (Mooney & DeJong, 1985) acquiring a schema for kidnapping by generalizing the explanation of a kidnapping narrative. The specification of the problem is given in Table 4. The schemata in the domain theory are defined by the set of lower-level schemata of which they are composed, and sets of preconditions, effects, and mental states which motivate the actor to perform the schema. The important goal achieved in the kidnapping narrative is John acquiring 250,000 dollars.

Table 4. Specification of the kidnap generalization problem.

Given:

- *Domain Theory:* Schemata for actions like BARGAIN, DRIVE, THREATEN, TELEPHONE, etc.. Inference rules relating states like FATHER(x,y) → PARENT(x,y).
- *Goal:* POSSESS(JOHN,$250,000).
- *Initial State & Observed Operator Sequence:* The Kidnap Story:
  Fred is the father of Mary and is a millionaire. John approached Mary. She was wearing blue jeans. John pointed a gun at her and told her he wanted her to get into his car. He drove her to his hotel and locked her in his room. John called Fred and told him John was holding Mary captive. John told Fred if Fred gave him $ 250,000 at Trenos then John would release Mary. Fred gave him the money and John released Mary.

Determine:

- A general schema for kidnapping.

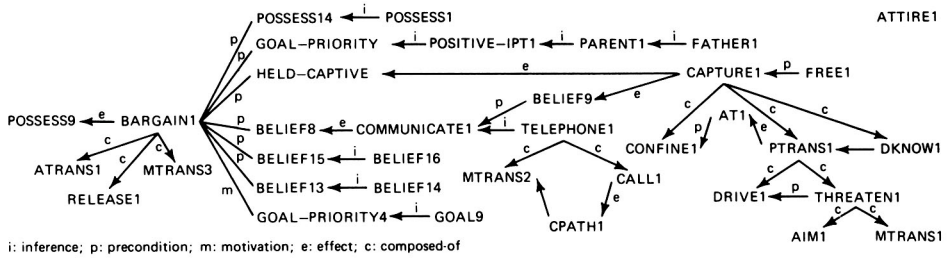

Figure 8. Complete causal structure of the kidnapping narrative. Interpretation of nodes shown in Table 5.

Figure 8 shows the causally complete structure the understanding process (step (1)) generates from the input text. The structure is instantiated to show the SPECIFIC interpretation. It shows how the low-level actions in the narrative can be interpreted as instances of general schemata the system already possesses, and how these schemata are causally connected to achieve the desired goal. As is evident from the figure, all the actions and states in the narrative support the achievement of the goal except the fact that Mary is wearing jeans. Consequently, only this fact is removed in step (2). The remaining structure is the explanation for how John acquired the money.

Step (3) eliminates nominal instantiations of schemata, or instantiations whose internal actions do not support the achievement of the ultimate goal except through the general goals achieved by the schema. This eliminates the specific actions John used to capture Mary and telephone Fred as well as the specific actions involved in the bargain. The resulting explanation is shown in Figure 9.

In step (4), actions and states which only support inferences to more abstract actions and states are retracted. This removes the state specifying that the ransom payer is the victim's father, since this state is subsumed by the more abstract state specifying that the ransom payer and victim have a close interpersonal relationship. This close interpersonal relationship is the only requirement in motivating the ransom payer to fulfill his end of the bargain. It also removes the specific action TELEPHONE since this action is subsumed by the more abstract action COMMUNICATE. The COMMUNICATE is the only requirement in transferring the appropriate information to the ransom payer which allows him to participate in the bargain.

Since the system does not have a more efficient method for achieving any of the subgoals in the plan, no optimization is done in step (5). Finally, applying the GENERAL substitution to the remaining structure in step (6) results in the final generalized explanation shown in Figure 10.

The generalized explanation is easily packaged into a schema by taking its leaves to be preconditions and the combined effects of its actions to be effects. This schema can then be stored away and used in understanding future narratives.

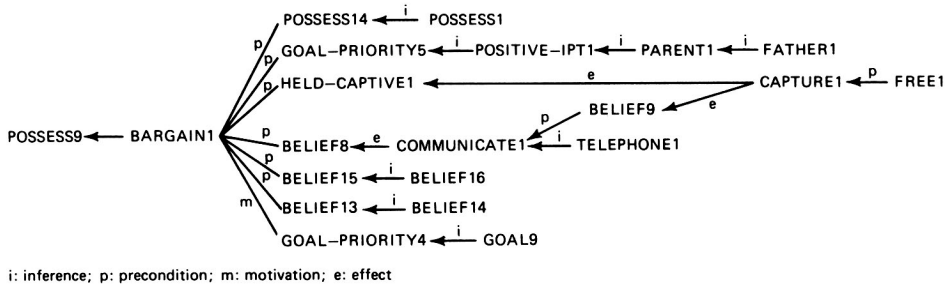

Figure 9. Explanation of kidnapping after Step 3. Interpretation of nodes shown in Table 5.

Table 5. Interpretation of nodes for Figures 8 & 9.

POSSESS9	John has $ 250,000.
BARGAIN1	John makes a bargain with Fred in which John releases Mary and Fred gives $ 250,000 to John.
MTRANS3	John tells Fred he will release Mary if he gives him $ 250,000.
RELEASE1	Fred releases Mary.
ATRANS1	Fred gives John $ 250,000.
POSSESS14	Fred has $ 250,000.
POSSESS1	Fred has millions of dollars.
GOAL-PRIORITY5	Fred wants Mary free more than he wants to have $ 250,000.
POSITIVE-IPT1	Fred has a positive interpersonal relationship with Mary.
PARENT1	Fred is Mary's parent.
FATHER1	Fred is Mary's father.
HELD-CAPTIVE1	John is holding Mary captive.
CAPTURE1	John captures Mary.
D-KNOW1	John finds out where Mary is.
PTRANS1	John moves Mary to his hotel room.
DRIVE1	John drives Mary to his hotel room.
THREATEN1	John threatens to shoot Mary unless she gets in his car.
AIM1	John aims a gun at Mary.
MTRANS1	John tells Mary he wants her to get in his car.
AT1	Mary is in John's hotel room.
CONFINE1	John locks Mary in his hotel room.
FREE1	Mary is free.
BELIEF8	Fred believes John is holding Mary captive.
COMMUNICATE1	John contacts Fred and tells him that he is holding Mary captive.
TELEPHONE1	John calls Fred and tells him that he is holding Mary captive.
CALL1	John called Fred on the telephone.
CPATH1	John had a path of communication to Fred.
MTRANS2	John told Fred he had Mary.
BELIEF9	John believes he is holding Mary captive.
BELIEF15	John believes Fred has $ 250,000.
BELIEF16	John believes Fred has millions of dollars.
BELIEF13	John believes Fred wants Mary to be free more than he wants to have $ 250,000.
BELIEF14	John believes Fred is Mary's father.
GOAL-PRIORITY4	John wants to have $ 250,000 more than he wants to hold Mary captive.
GOAL9	John wants to have $ 250,000.
ATTIRE1	Mary is wearing blue jeans.

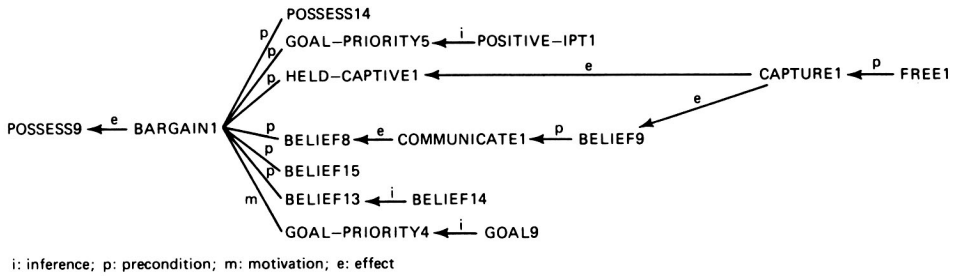

Figure 10. Final generalized explanation of kidnapping. Interpretation of nodes shown in Table 6.

Table 6. Interpretation of nodes for Figure 10.

POSSESS9	Person1 has Money1.
BARGAIN1	Person1 makes a bargain with Person2 in which Person1 releases Person3 and Person2 gives Money1 to Person1.
POSSESS14	Person2 has Money1.
GOAL-PRIORITY5	Person2 wants Person3 free more than he wants to have Money1.
POSITIVE-IPT1	There is a positive interpersonal relationship between Person2 and Person3.
HELD-CAPTIVE1	Person1 is holding Person3 captive.
CAPTURE1	Person1 captures Person3.
FREE1	Person3 is free.
BELIEF8	Person2 believes Person1 is holding Person3 captive.
COMMUNICATE1	Person1 contacts Person2 and tells him that he is holding Person3 captive.
BELIEF9	Person1 believes he is holding Person3 captive.
BELIEF15	Person1 believes Person2 has Money1.
BELIEF13	Person1 believes Person2 wants Person3 to be free more than he wants to have Money1.
BELIEF14	Person1 believes there is a positive interpersonal relationship between Person2 and Person3.
GOAL-PRIORITY4	Person1 wants to have Money1 more than he wants to hold Person3 captive.
GOAL9	Person1 wants to have Money1.

10. Conclusions

In this paper our approach to learning has been contrasted at some length with the approach of Mitchell, Keller, and Kedar-Cabelli (1986). One might forget that in the universe of learning research our approaches are very similar. Far more views and biases unite us than divide us. Indeed, this is true across most explanation-based learning efforts. Let us contrast the explanation-based approach with generalization learning in a few non-explanation-based systems.

In the ACT system (Anderson, 1983), processing is done in a production system framework. The production rules formulate solutions to problems given to the

system. Generalized rules are formed to capture the generalities among two examples. Each example consists of a problem and its solution. The generalization captures what the two examples have in common. The generalization process does not itself interact with any domain knowledge. It is performed by a partial pattern matcher, incidentally the same partial pattern matcher that underlies analogical learning. It can be contrasted to the explanation-based approach in two ways: 1) there is no domain theory that participates in the generalization process, 2) the generalization algorithm is well defined only with multiple examples.

The SOAR system (Laird *et al.*, 1984), shares the notion of knowledge chunking with our work and indeed with much of the other explanation-based work. Their claim is that chunking of knowledge (in the form of production rules) is universal for learning. That is, no other learning mechanisms need be postulated. In particular there is no domain theory needed to guide the generalization process. This, again, differs from the explanation-based learning work. Furthermore, no researcher in explanation-based learning has yet made a claim of universality for the approach; most researchers seem to believe that the ultimate answer to learning will involve a combination of explanation-based, similarity based and analogical algorithms.

EURISKO (Lenat & Brown, 1984) is in some sense the dual to explanation-based systems. Both are knowledge based and require some form of domain theory to be input. However, EURISKO attempts to discover heuristic concepts. Heuristics are rules of thumb that seem to hold in the domain but for which there is no explanatory analysis. EURISKO tries to discover helpful concepts through massive simulation. An explanation-based system, on the other hand, generates a concept only if it can analyze why it works.

The explanation-based approach to learning holds great promise, particularly in the area of learning apprentice systems (Minton, 1985; Mitchell *et al.*, 1985; O'Rorke, 1984; Wilkins *et al.*, 1985) and other systems that can learn from observing behavior beyond their capabilities (Ellman, 1985; Minton, 1984; Mooney & DeJong, 1985; Segre & DeJong, 1985; Shavlik, 1985). However, recent research in explanation-based learning has only scratched the surface.

To date, working explanation-based systems have learned by explaining new configurations of known operators. We call this *schema composition.* There are many other uses of explanations to drive both generalization and concept refinement (DeJong, 1983).

Another important area for future research is combining similarity-based and explanation-based methods. There has already been significant research on combining the two approaches along many different directions (Lebowitz, 1985; Mitchell, 1984; Pazzani, 1985; Porter & Kibler, 1985; Rajamoney *et al.*, 1985).

Finally, we wish to stress the important part that implementation of large computer systems played in our work. Time and again algorithms that appeared elegant on paper could not be implemented. Even more important were the gaps in our theories that attempts at implementation exposed. Theories that appeared to be

general and worked well in several domains failed to support an apparently similar type of concept acquisition when extended to yet another domain. We believe that significant implementations in many diverse domains, although often painful and in many ways expensive, is a necessary step toward a formalized general theory of explanation-based learning.

Acknowledgements

The authors benefited greatly from discussions with other members of the explanation-based learning group at Illinois: Paul O'Rorke, Alberto Segre, Jude Shavlik, Shankar Rajamoney, Scott Bennett, Steve Chien, and Paul Chen. Copious comments by Pat Langley also helped to improve the clarity of our work. Finally we wish to thank Tom Mitchell, Rich Keller, and Smadar Kedar-Cabelli who functioned admirably serving in the somewhat awkward position of both reviewer and reviewee. This work was supported by the National Science Foundation under grant NSF-IST-83-17899.

References

Anderson, J.R. (1983). Acquisition of proof skills in geometry. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (pp. 191–221). Palo Alto, CA: Tioga.

Carbonell, J.G., Michalski, R.S., & Mitchell, T.M. (1983). An overview of machine learning. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (pp. 331–363). Palo Alto, CA: Tioga.

Chafe, W. (1974). Some thoughts on schemata. *Theoretical issues in natural language processing, 1*, 89–91.

Charniak, E. (1976). A framed painting: The representation of a common sense knowledge fragment. *Cognitive Science, 4*, 355–394.

DeJong, G.F. (1981, August). Generalizations based on explanations. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence* (pp. 67–70). Vancouver, British Columbia, Canada: Morgan-Kaufmann (Also appears as Working Paper 30. AI Research Group. Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.)

DeJong, G.F. (1983, June). An approach to learning from observation. *Proceedings of the Third International Machine Learning Workshop*. Urbana, IL. (Also appears as Working Paper 45, AI Research Group. Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.)

DeJong, G.F. (1983, August). Acquiring schemata through understanding and generalizing plans. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*. Karlsruhe, West Germany: Morgan-Kaufmann (Also appears as Working Paper 56, AI Research Group. Coordinated Science Laboratory. University of Illinois at Urbana-Champaign.)

DeJong, G.F. (1986). Explanation based learning. In R.S. Michalski, J.G. Carbonell, T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach Volume II*. Los Altos, CA: Morgan Kaufmann.

Ellman, T. (1985, August). Generalizing logic circuit designs by analyzing proofs of correctness. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 643–646). Los Angeles, CA: Morgan-Kaufmann.

Fikes, R.E., Hart, P.E., & Nilsson, N.J. (1972). Learning and executing generalized robot plans. *Artificial Intelligence, 3*, 251–288.

Hofstadter, D. (1983). Personal communication.

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

Lebowitz, M. (1985, June). Complex learning environments: Hierarchies and the use of explanation. *Proceedings of the Third International Machine Learning Workshop* (pp. 110–112). Skytop, PA.

Lenat, D.B., & Brown, J.S. (1984). Why AM and EURISKO appear to work. *Artificial Intelligence, 23*, 269–294.

Mahadevan, S. (1985, August). Verification-based learning: A generalization strategy for inferring problem-reduction methods. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 616–623). Los Angeles, CA: Morgan-Kaufmann.

Minsky, M.L. (1975). A framework for representing knowledge. In P. Winston (Ed.), *The psychology of computer vision* (pp. 211–277). New York: McGraw-Hill.

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

Minton, S. (1985, June). Overview of the PRODIGY learning apprentice. *Proceedings of the Third International Machine Learning Workshop* (pp. 120–122). Skytop, PA.

Minton, S. (1985, August). Selectively generalizing plans for problem-solving. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 596–599). Los Angeles, CA: Morgan-Kaufmann.

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

Mitchell, T. (1984). Toward combining empirical and analytic methods for learning heuristics. In A. Elithorn & R. Banergi (Eds.), *Human and artificial intelligence*. Amsterdam: North Holland.

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

Mitchell, T.M., Keller, R., & Kedar-Cabelli, S. (1986). Explanation-based generalization: A unifying view. *Machine learning, 1*, 47–80.

Mooney, R., & DeJong, G.F. (1985, August). Learning schemata for natural language processing. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*. Los Angeles, CA: Morgan-Kaufmann. (Also appears as Working Paper 67. AI Research Group. Coordinated Science Laboratory. University of Illinois at Urbana-Champaign.)

Mooney, R., & Bennett, S. (1986, January). *A domain independent explanation-based generalizer*. Working Paper 71. AI Research Group, Coordinated Science Laboratory, University of Illinois, Urbana, IL.

Mostow, D. (1983). Machine transformation of advice into a heuristic search procedure. In R. Michalski, J.G. Carbonell, T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (pp. 367–404). Palo Alto, CA: Tioga.

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

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

Pazzani, M. (1985, August). Explanation and generalization based memory. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society* (pp. 323–328). Irvine, CA: Morgan-Kaufmann.

Porter, B., & Kibler, D. (1985, August). A comparison of analytic and experimental goal regression for machine learning. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 555–559). Los Angeles, CA: Morgan-Kaufmann.

Rajamoney, S., DeJong, G.F., & Faltings, B. (1985, August). Towards a model of conceptual knowledge acquisition through directed experimentation. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*. Los Angeles, CA: Morgan-Kaufmann. (Also appears as Working Paper 68, AI Research Group. Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.)

Sacerdoti, E. (1974). Planning in a hierarchy of abstraction spaces. *Artificial Intelligence, 5*, 115–135.

Schank, R. & Abelson, E. (1977). *Scripts, plans, goals and understandings: An inquiry into human knowledge structures*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Schank, R.C. (1982). *Dynamic memory*. Cambridge, England: Cambridge University Press.

Segre, A.M., & DeJong, G.F. (1985, March). Explanation based manipulator learning: Acquisition of planning ability through observation. *Proceedings of the IEEE International Conference on Robotics and Automation* (pp. 555–560). St. Louis, MO. (Also appears as Working Paper 62, AI Research Group, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.)

Shavlik, J.W. (1985, August). Learning about momentum conservation. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 667–669). Los Angeles, CA: Morgan-Kaufmann. (Also appears as Working Paper 66, AI Research Group, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.)

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

Waldinger, R. (1977). Achieving several goals simultaneously. In E. Elcock & D. Michie (Eds.), *Machine Intelligence, 8*.

Wilkins, D., Clancey, W., & Buchanan, B. (1985, June). ODYSSEUS: A learning apprentice. *Proceedings of the International Machine Learning Workshop* (pp. 221–223). Skytop, PA.

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