Why and How to Learn Why: Analysis-based Generalization of Procedures

CLAYTON LEWIS
University of Colorado

Max Wertheimer, in his classic Productive Thinking, linked understanding to transfer: Understanding is important because it provides the ability to generalize the solution of one problem to apply to another. Recent work in human and machine learning has led to the development of a new class of generalization mechanism, called here analysis-based generalization, which can be used to provide a concrete account of the linkage Wertheimer suggested: these mechanisms all, in different ways, use understanding of examples in the generalization process. In this paper I review this class of mechanism, and describe a method for causal attribution that can produce the analyses of examples that the generalization methods require, in the domain of simple procedures in human-computer interaction. This causal analysis method is linked with analysis-based generalization to form EXPL, an implemented model which is a concrete, though limited, instantiation of Wertheimer's scheme. EXPL constructs an understanding of an example procedure and generalizes it on the basis of that understanding. Results of an empirical study suggest that some of EXPL's attribution heuristics are used by people, and that while a subclass of analysis-based methods, called superstition methods, seem to provide a more plausible account of people's generalization under the conditions of the study than a contrasting class of rationalistic methods, at least some participants appear to use methods from both classes. The results also show that explanation-based methods, which rely on comprehensive domain theories, must be used in conjunction with a means for extending the domain theory. If thus enhanced, explanation-based methods are able to mimic the effects of other analysis-based methods, and can provide a good account of the data, though combinations of other methods must also be considered. Finally, I return to Wertheimer's ideas to argue that none of the current analysis-based generalization methods fully captures Wertheimer's notion of understanding. Proper choice among different possible analyses of an example is crucial for Wertheimer, but I argue that this problem may be beyond the reach of learning systems.

I thank Mitchell Blake, Stephen Casner, and Victor Schoenberg for their assistance in the research described here. Many others have been generous with ideas and suggestions, including Richard Alterman, John Anderson, Susan Bovair, Gary Bradshaw, Lindley Darden, Steven Draper, David Kieras, Donald Norman, Peter Polson, Jonathan Shultis, and Ross Thompson, James Greeno, Raymond Mooney, Gary Olson, and Peter Pirolli suggested numerous improvements on an earlier version. This work was supported by the Office of Naval Research, Contract No. N00014-85-K-0452, with additional contributions from the Institute of Cognitive Science and AT&T.

Send correspondence and requests for reprints to Clayton Lewis, Department of Computer Science and Institute of Cognitive Science, Campus Box 430, University of Colorado, Boulder, CO 80309.
INTRODUCTION

What is the point of understanding something, rather than simply knowing it? This is a crucial question in cognitive science and its applications. Wertheimer's classic *Productive Thinking* (1959, originally published in 1945) traced the boundary between understanding and not understanding in a series of examples including the famous problem of finding the area of a parallelogram. Wertheimer argued that understanding consists in grasping the *inner structural relationships* in a problem. In the example of the parallelogram, the crux of the problem, for Wertheimer, is seeing that moving a piece of the figure from one end to the other disposes of two discrepancies between the parallelogram and a rectangle, whose area can be determined.

But why are such insightful solutions valuable? While Wertheimer does not address this question directly, it is implicit in his discussion that the reward is generalization: solutions that embody understanding can be extended to a wider range of new problems than solutions which do not. Thus, the insightful solution to the parallelogram problem can be extended to find the area of a trapezoid and other even less regular figures, while the formula itself, or even a construction justifying the formula, is not transferable.

Recent work on generalization, in psychology and artificial intelligence, is moving onto the same ground explored by Wertheimer, from the opposite direction. While Wertheimer was interested in characterizing understanding, and only secondarily in generalization, this new work aims to produce generalizations, and is only secondarily interested in understanding. The work is converging with Wertheimer's because of the rediscovery that generalization and understanding are linked. Since the new work is focused on mechanisms of generalization it provides clearer and more concrete ideas than Wertheimer could of the way in which understanding aids in generalization. Complementarily, it appears that Wertheimer's view of understanding picks out some issues that have not been addressed in the recent work.

Plan of the Paper

The discussion will focus on generalization of procedures, so I begin by describing how this special case of generalization relates to the abstract problem of characterizing a set of objects given examples from the set. I then review a class of generalization mechanism which I call *analysis-based*, in which generalizations are based on an analysis, or understanding, of a single example, rather than on a description of a large class of examples, as in inductive methods. The class includes explanation-based methods, analogical generalization, and a method in which new procedures are synthesized from operator definitions gleaned from the analysis of an example. Within the class I distinguish *superstitious* methods, in which aspects of an example that are not understood are retained in generalizations built from it, from *rationalistic* methods, in which only aspects that are understood are used in generalizations.
I next describe a method for causal attribution that can produce the analyses of examples that the generalization methods require, in the domain of simple procedures in human-computer interaction. This causal analysis method is linked with either of two analysis-based generalization methods to form EXPL, an implemented model which is a concrete, though limited, instantiation of Wertheimer's scheme. EXPL constructs an understanding of an example procedure and generalizes it on the basis of that understanding.

I next report the results of a study aimed at determining how good an account of people's analysis of examples is provided by EXPL's causal attribution heuristics, and which (if any) analysis-based generalization methods may be involved in people's generalizations. The results suggest that some of EXPL's attribution heuristics are used by people, and that while the subclass of superstitious methods seem to provide a more plausible account of people's generalization under the conditions of the study than the rationalistic subclass, at least some participants appear to use methods from both classes. The results also show that people readily generalize about examples about which they have very limited background knowledge, so that explanation-based methods, which rely on comprehensive domain theories, must be used in conjunction with a means for extending the domain theory in processing a given example.

In the general discussion I first consider the generality of EXPL's causal attribution heuristics. Turning to the findings on generalization I show how explanation-based methods can be extended to handle material for which a prior domain is lacking. If this is done, explanation-based methods are able to mimic the effects of other analysis-based methods, producing either superstitious or rationalistic generalizations, the different behaviors being governed by the character of the domain theory employed. With this extension explanation-based models can provide a good account of the data, though combinations of other methods must also be considered. Finally, I return to Wertheimer's ideas to argue that none of the current analysis-based generalization methods fully captures Wertheimer's notion of understanding. In particular, proper choice among different possible analyses of an example is crucial for Wertheimer. I argue further that Wertheimer's suggestions about how to make this choice are not workable, and that this problem may be beyond the reach of learning systems.

ANALYSIS-BASED GENERALIZATION METHODS

The work on generalization to be reviewed here has appeared under the headings "explanation based learning" (DeJong, 1981, 1983a; DeJong & Mooney, 1986; Kedar-Cabelli, 1985; Mitchel, Keller, & Kedar-Cabelli, 1986) "analogical generalization" (Anderson & Thompson, 1986; Gentner, 1983; Pirolli, 1985) and "human-computer interaction" (Lewis, 1986a,b). In all of the approaches to be described, in contrast with earlier "similarity-based"
or "inductive" methods which look for regularities among large numbers of examples (for review see Dietterich & Michalski, 1983), generalizations are based on an analysis of one or a few examples. The analysis aims to determine why an example is an example, so that further examples can be recognized or constructed.

I will discuss the application of these analysis-based generalization methods in a single task domain: generalizing simple procedures in human-computer interaction. There has been some success in modelling the process by which examples are analyzed in this domain, while this necessary precursor to generalization has not been examined in depth in other domains. The availability of a model of the analysis process, together with the various generalization techniques which can use an analysis, permits me to assemble a complete model of the analysis and generalization process, whose feasibility can be tested.

**The Generalization Problem for Procedures**

Formally, the solution to a generalization problem is a characterization of a set of objects given examples drawn from the set (and sometimes nonexamples). In a procedural domain the objects of interest are procedure-outcome pairs, and the set to be characterized is the set of procedure-outcome pairs associated with some context of execution. In human-computer interaction what is wanted is a description of how procedures and outcomes are paired by the particular system being used, that is, that pairs procedures with the outcomes that would be produced if they were executed on the system. Of special interest are descriptions of the pairing that make it possible to determine a procedure that will produce a given outcome, if one exists.

In inductive approaches generalizations are developed by examining a number of examples of a to-be-learned concept and constructing an economical description that is satisfied by all the examples (and not by any known nonexamples). The generalization produced is the conjecture that any item that satisfies this description is a member of the concept.

Analysis-based approaches attempt to build generalizations not by characterizing a number of examples but by discerning the essential features of a single example. By explaining what makes this example an example, we can characterize a larger class of examples, namely the class of examples for which the same explanation holds.

Some of the methods I will discuss use the analysis of an example to produce an explicit generalization, while in others the generalization is implicit. The former methods provide a description of a class of procedure-outcome pairs; to build a procedure that accomplishes a given outcome it is necessary to use the description to characterize the procedure or procedures that are paired with the desired outcome. The latter methods provide no explicit description of the class of procedure-outcome pairs. Instead, they use the
analysis of an example, and a desired outcome, to construct a procedure that is paired with that outcome. This process implicitly defines a class of procedure-outcome pairs.

Explanation-based Approaches

**Explanation-based Generalization (EBG).** Mitchell et al. (1986) describe an analysis-based technique, called EBG, in which the analysis of an example consists of a proof, within a formal theory of the example domain, that the example belongs to a specified goal concept. The generalization process examines this proof and constructs an explicit characterization of the class of examples for which essentially the same proof would work. In contrast to similarity-based generalizations, a generalization constructed in this way can be formally proven to be correct, even though it may be based on only one example.

**Explanation-based Learning (EBL).** DeJong and Mooney (1986) discuss a broader framework, called explanation-based learning, in which the analysis of an example is embodied in a set of interlocking schemata which the example instantiates and which account for the aspects of the example that are to be understood. In a procedural domain the schemata fit to an example would pick out the causal links between the procedure and its outcome. Just as EBG generalizes to the class of examples for which a given proof would go through, explanation-based learning generalizes to the class of examples to which a given schema or collection of schemata can be fit.

**Dependence on Domain Theory.** Both EBG and EBL require a domain theory to be given, which is unavailable in many realistic learning contexts, as Kedar-Cabelli (1985) and Mitchell et al. (1986) note. In EBG this theory is a set of rules and facts which must be capable of supporting a proof that the example is a member of the to-be-learned concept. In EBL the theory consists of a collection of schemata which must be adequate to cover the example, in the sense that there must be schemata in the domain theory which can be fit to all of the essential parts of the example and which account for the roles the parts play in the example as a whole.

In the domain being considered here, procedures for operating computers, learners frequently encounter examples that they cannot explain on the basis of prior knowledge, that is, examples for which they do not possess an adequate domain theory. Command names provide a simple example of this difficulty. In some operating systems "dir" is a command for displaying a directory of files. When a learner first encounters this command he or she would probably not know this. Thus, when an example using "dir" is first encountered, say in a demonstration, the learner’s domain theory is in-
adequate to prove that the procedure in the example accomplishes the observed outcome (as required in EBG), or to provide a schema to be matched to the example which links "dir" with the observed effect, as required in EBL. But it seems probable that as a result of seeing an example of the use of "dir," the learner can readily grasp what "dir" does, and augment his or her knowledge accordingly. It appears in cases like this that extending the domain theory to account for new examples is a key process in generalization, one not encompassed by EBG or EBL. I will return to this issue, and what might be done about it, after determining whether learners are actually able to generalize in the absence of adequate background knowledge.

**Analogical Generalization**

*Structure Mapping.* Given a procedure P, its outcome O, and some new outcome O', we can form an analogy involving a new, unknown procedure, X, as follows:

\[ P : O :: X : O' \]

If we have an analysis describing why P produces O, which picks out particular relationships between the parts of P and aspects of O, we can use *structure mapping* (Gentner, 1983) and try to impose these same relationships on X and O'. As the name suggests, having determined what we think is the important structure in the P : O pair we map that structure across the analogy and impose it on the X : O' pair. In favorable cases this structure, which is represented as a collection of relationships that must hold between X and O', will constrain X enough that we can construct it.

Here is a simple example. Suppose the procedure TYPE 'DELETE,' TYPE 'EGGPLANT' removes the file EGGPLANT. Our analysis of this procedure and its outcome might indicate that the name of a file in the outcome must also appear as the second step of the procedure. If a different file appears in O', the desired new outcome, we can satisfy this relationship by including in the new procedure X a step mentioning the name of the new file.

*Analogical Generalization in PUPS.* Another approach to dealing with the above analogy is to rearrange it as follows:

\[ O : O' :: P : X \]

If we can find a transformation that maps O to O' we expect that the same transformation should change P into X. Anderson and Thompson's PUPS system (Anderson & Thompson, 1986) works this way; similar ideas are discussed in Pirolli (1985) and Dershowitz (1986). I will follow PUPS in our discussion, and will use that name to refer to the approach. The reader should be aware, however, that the PUPS system contains many elements which I judge not to be central to this discussion, including the construction
of production rules that encode generalizations, the use of spreading activation to select appropriate examples to generalize for a given purpose, a discrimination mechanism to deal with overgeneralizations, and others. My use of the name PUPS refers only to its method of constructing generalizations. Anderson (1987) describes how some of the additional features of PUPS are used in learning procedures in algebra, an application similar to the one I am describing here.

As applied to our domain, a to-be-generalized example in PUPS consists of a procedure, a description of its outcome, and indications of the roles played by the parts of the procedure in producing the outcome. Given a new outcome a simple substitution mapping is constructed that transforms the old outcome into the new one. This mapping is then applied to the parts of the old procedure, given a new procedure that (it is hoped) produces the new outcome.

Here is a simple example. Suppose the procedure TYPE "DELETE," TYPE "EGGPLANT" removes the file named EGGPLANT from a system. How would we remove the file BROCCOLI? In mapping the old outcome to the new one we need only replace EGGPLANT by BROCCOLI. Applying this same replacement to the command we get the new procedure TYPE "DELETE," TYPE "BROCCOLI." This example is trivial, in that we did not need any information about the roles of parts of the procedure.

Now suppose we wish to accomplish the new goal of printing the file EGGPLANT. Suppose further that in addition to the knowledge that TYPE "DELETE," TYPE "EGGPLANT" removes the file EGGPLANT, we know these facts: "DELETE is the command for removing" and "WRITE is the command for printing." Mapping the old outcome, removing the file EGGPLANT, to the new outcome is accomplished by replacing "removing" by "printing." In contrast to the first example, the term "removing" does not appear in the to-be-modified procedure, so we seem to be stuck. We can't just replace "removing" by "printing" because "removing" does not appear in the procedure we are trying to modify.

The PUPS process gets around this impasse by examining the roles of the parts of the procedure. Finding that the role of DELETE is "the command for removing," it applies the mapping to this role, obtaining "the command for printing." It then looks for an implementation of this modified role, obtaining WRITE. It then substitutes WRITE for DELETE, obtaining TYPE "WRITE," TYPE "EGGPLANT." Note that PUPS' ability to solve this problem depends crucially on an analysis that tells it what the role of DELETE is in the example.

Contrast Between Structure Mapping and PUPS. Structure mapping and PUPS have in common the exploitation of the idea of analogy, and the dependence on an analysis of how a to-be-generalized example works. Structure mapping embodies this analysis in the structure that is attributed to P
and O, and that is then imposed on X and O'. PUPS embodies the analysis in the assignment of the roles that are used to guide the modification process. But the two methods differ in their treatment of unanalyzed aspects of examples, an issue which will be important below. Structure mapping only imposes on the new procedure X those constraints which it has discerned in P and O; any aspects of P that were not implicated in the analysis of its relationship to O will not be mapped over to X and O', and hence will not be reflected in X. By contrast, any aspect of P that is not assigned a role in PUPS will be left unchanged by the modification process, and will survive in the X, the result of modifying P.

**Justifiability of Generalizations in Analogical Generalization.** Analogical generalization resembles EBG and EBL in that it can extract the information needed to support a generalization from a single example, and requires an analysis of how the example works, rather than just a description of it. But unlike explanation-based generalizations those based on analogies may be invalid. For example, in the case last discussed it could be that DELETE only works with files whose names begin with E. This possibility does not occur in EBG because of the requirement for a formal domain theory in which membership in a concept can be rigorously proved, or in EBL, providing the schemata in the domain theory are correct. Analogical generalization requires no comprehensive domain theory and pays a price for it.

Russell (1987) points out that analogical generalization could be rigorously grounded by requiring that any analogy be backed by a domain theory which asserts that the known common properties of a given example and a proposed new example, on which the analogy is based, are logically sufficient to determine that they share other properties which are known for the given example but not for the new. For example, if a domain theory includes the assertion that the first word in a command logically determines the operation performed by the command, then generalizing about operations from an example containing a particular first word, like DELETE, to another command with the same first word, must be safe. The two commands starting with DELETE must produce the same operation, which the given example shows must be removing, regardless of what the operand is, if the domain theory is correct. It can't happen, under this domain theory, that DELETE only works on files whose names start with E.

It is not clear whether Russell's idea offers an appropriate addition to structure mapping or PUPS, if these mechanisms are considered as psychological models. Just as it seems probable that learners can generalize about commands they have never seen, and for which they therefore lack a prior theory, it seems that they can use analogy productively in situations in which they lack the domain knowledge necessary to derive the assertions about what determines what that Russell's scheme relies on. Pirolli and Anderson (1985) describe the use of analogy by a LOGO learner who clearly could not
justly her analogy rigorously given her incomplete understanding of the language. It is possible, however, that learners do make conjectures about what determines what, that these conjectures play the role Russell outlines, and that the formation of such conjectures plays an important role in their use of analogy.

Synthetic Generalization

In earlier work on the role of explanations in learning (Lewis, 1986a) I developed a generalization technique that resembles structure mapping and PUPS in not requiring a formal domain theory, but that produces new procedures by building them out of small, separately-understood parts rather than by modifying an example, as in PUPS, or by mimicking the structure of an example, as in structure mapping. Richard Alterman (personal communication) calls this distinction the "little chunk - big chunk" contrast in the context of planning systems. A "big chunk" planner works by finding a known plan that accomplishes roughly what is needed, and the modifying it as required. A "little chunk" planner works from a repertoire of small steps whose behavior it knows. Faced with a novel goal, it builds a procedure to accomplish it from scratch, using these primitive steps.

Synthetic generalization works as follows on the TYPE "DELETE," TYPE "EGGPLANT" example. Assume that an analysis of the example has yielded the information that TYPE "DELETE" specifies a removal operation, and that TYPE "EGGPLANT" specifies the indicated file. Suppose that analysis of a second example reveals that TYPE "WRITE" specifies a print operation (say) and that TYPE "BROCCOLI" specifies the file BROCCOLI. The examples themselves are discarded; only the information about primitive pieces is retained. Given the demand to remove BROCCOLI, synthetic generalization builds the procedure TYPE "DELETE," TYPE "BROCCOLI" by putting together TYPE "DELETE" and TYPE "BROCCOLI."

The principles underlying synthetic generalization are very close to those underlying the work of Winston and colleagues on learning physical descriptions for objects with functional definitions (Winston, 1980, 1982; Winston, Binford, Katz, & Lowry, 1983). Winston et al. use auxiliary examples, called precedents, to establish connections between physical features and functional properties; these connections correspond to the connections in synthetic generalization between pieces of a procedure and aspects of its outcome. Because of the goal of recognizing objects rather than constructing them the Winston work does not build collections of features, as would synthetic generalization, but rather constructs efficient recognition rules for constellations of features that might be observed in other examples.

The Winston work presupposes simple relationships between features and functional properties, so that features that determine properties like 'liftable' and 'stable' in separate precedents can just be combined when
recognizing a single object that is both liftable and stable. If liftability and stability interacted, so that liftability was determined by different features when an object is stable and when it is not, this simple combination scheme would fail. In the same way, synthetic generalization presupposes that the roles of individual steps, gleaned from the analysis of separate examples, can be used to predict in a simple way what will happen when these steps are combined.

This background assumption about combinability of steps is not required in structure mapping or PUPS. The critical relationships between a procedure and its outcome that structure mapping enforces can be arbitrary: they could take into account complex interactions between steps, if necessary, though determining what these interactions are when analyzing an example might not be easy. Similarly PUPS can deal with complex interactions among parts of an example by not assigning roles to small parts but only to groups of parts.

**Rationalism versus Superstition**

A key point about synthetic generalization is that it might produce the procedure TYPE "BROCCOLI," TYPE "DELETE" rather than TYPE "DELETE," TYPE "BROCCOLI" in the example considered above. Its knowledge about DELETE and filenames does not include anything about the order in which steps involving them must occur, and the synthetic generalization procedure does not have access to the original examples from which its knowledge was derived. By contrast, PUPS will rarely reorder an example, because a new procedure is always obtained by substituting parts in the example. Only when a substitution interchanges parts, or an example is originally described as containing unordered steps, both unusual circumstances, would reordering occur.

A similar contrast emerges in the treatment of unexplained parts of a procedure. In synthetic generalization, a step that is seen in an example, but whose role is mysterious, will never be included in a new procedure, because the synthesizer will have no description of its effects. In PUPS an unexplained part of the procedure, that is, one that has no role, will be left unchanged in the modification process.

Let us call synthetic generalization a *rationalistic* process, in that generalizations include only features of examples, such as order or particular steps, whose action is understood, and PUPS a *superstitious* process, in that features of examples that are not understood are carried forward into generalizations. Under this definition structure mapping is a rationalistic process, for reasons discussed above: Parts of a procedure that do not participate in known relationships with its outcome will not be reproduced in the generalized procedure.

We might expect superstitious generalization to be important in complex, poorly-understood domains. Rationalistic generalization will not perform
well when a complete analysis of how an example works is not available. On the other hand, rationalistic generalization might be more useful when examples are hard to remember in detail, or when new problems are not very close to the examples one has seen.

Explicit versus Implicit Generalization
As noted earlier, some of these methods produce explicit generalizations, that is, descriptions of the desired class of procedure-outcome pairs, while others do not. In particular, EBG and EBL produce such descriptions, in the form of a predicate, in EBG, or a generalized schema, in EBL. The other methods produce only implicit generalizations: they will construct a new pair in the class only when given a desired outcome.

ANALYSIS OF EXAMPLES
All of the generalization methods just described need the same kind of information, packaged in different ways, about an example procedure and its outcome: what parts of the procedure cause what aspects of the outcome. To build a complete model of Wertheimer’s framework we need a process that can provide these causal attributions. How might such a process work?

Thinking-aloud studies of people learning to use computers (Lewis & Mack, 1982; Mack, Lewis, & Carroll, 1983) provide a couple of suggestions. First, learners seemed to pay attention to coincidences, or identities, between elements of their actions and elements of results. For example, one learner conjectured that a message containing the word FILE was the outcome of a command containing the word FILE, though in fact the message was unrelated to the command and the occurrence of FILE in both was a coincidence. Second, faced with examples containing multiple actions and results learners appeared to partition results among actions in such a way that a single action was presumed to have produced a single result. These cases suggested that learners may possess a collection of heuristics that enable them to conjecture the relationships among actions and outcomes in a procedure. Here are descriptions of two candidate heuristics.

The Identity Heuristic. Suppose that we are watching a demonstration of an unfamiliar graphics editor. After a series of actions which we do not understand, the demonstrator draws a box around an object on the screen. After some further uninterpretable actions the object in the box disappears. We might conjecture that the drawing of the box specified the object that was to disappear; that is, that the earlier user action of drawing the box around the object was causally connected with the later system response involving the identical object. This heuristic, which ties together actions and responses that share elements, is reminiscent of the similarity cue in causal
attribution (Shultz & Ravinsky, 1977), in which causes and effects which are similar in some respect may be linked.

The Loose-ends Heuristic. Suppose in watching another demonstration we are able to explain all but one user action and all but one system response, which occurs later. We might conjecture that the otherwise unexplained action is causally linked to the otherwise unexplained response. We might justify our conjecture with two assumptions: that a demonstration shows an economical way to accomplish its outcome and that all aspects of system responses are attributable to some user action.

This heuristic captures some of the observed partitioning of results among actions by learners mentioned above. It is consistent with the "determinism" assumption discussed in the causal attribution literature (Bullock, Gelman, & Baillargeon, 1982), by which all events are assumed to have causes.

THE EXPL SYSTEM

The EXPL system (Lewis, 1986a) was developed to explore these and similar heuristics, and their role in generalization. It implements a small set of heuristics in such a way as to produce the information required by PUPS or synthetic generalization from an example. It combines this causal analysis with PUPS or with synthetic generalization, providing a complete model of procedural learning from examples, in which extracting information from examples, and use of that information to produce new procedures, are both represented. EXPL thus provides a feasibility demonstration that these analysis-based methods, together with causal analysis, can perform generalization of procedures. There appears to be no reason why the analysis EXPL produces could not drive structure mapping as well, but this has not been done. I will discuss in the following sections those aspects of EXPL pertinent to the examples considered in this paper; complications and extensions needed to handle some more complex examples are described in Lewis (1986a).

Encoding
Examples are represented to EXPL as a series of events, each of which is either a user action or a system response. An event is made up of one or more components, which may represent objects, commands, operations, or other entities. These components are treated by EXPL as arbitrary, uninterpreted tokens, with a few exceptions that need not be considered here. No significance attaches to the order in which components of an event are listed. Figure 1 shows an example as described in English and as encoded for EXPL.

This primitive encoding scheme has many limitations; it cannot represent relationships among entities within an event, such as the information that a
User types letter 'd' on keyboard.
User touches picture of train on screen.
System removes train from screen.

Example as encoded for EXPL:

```
u type d
u touch train
s remove train
```

Figure 1. Example of procedure and outcome.

collection of entities all appear on the same menu, for example. But it has proved adequate to support the analysis of examples of modest complexity and it is sufficient to support the implementation of the EXPL analysis heuristics which are our focus here.

**The Identity Heuristic in EXPL.** When a component of a system response has occurred earlier in a user action, EXPL asserts that that user action specified that component of the system response. For example, if clicking a mouse on an object is followed by the disappearance of that object, EXPL asserts that it was clicking on the object that led to that object, rather than some other, disappearing.

EXPL's implementation relies on the encoding process to enable the identity heuristic to be applied in some cases. Suppose a picture of an object disappears after the name of the object is mentioned. The encoding of these events must use the same token to represent the picture and the name. Otherwise the identity heuristic will be unable to link the mention to the disappearance. A more sophisticated implementation would permit encodings with multiple descriptions of events, and use background knowledge to link tokens which are not identical but have related meanings. EXPL's primitive approach is adequate to support our discussion, however.

**The Obligatory Previous Action Heuristic.** EXPL's analysis assumes that system responses occur rapidly with respect to the pace of user actions, so that system responses will occur as soon as all contributing user actions have been made. Consequently, some contribution from the immediately previous user action must always be posited.

**The Loose-ends Heuristic.** If EXPL finds a user action which it cannot connect to the goal of an example, and it finds a component of a later system response that it cannot account for, it posits that the unexplained user
action is linked to the unexplained system response. In the current system
the goal of an example is identified with the final system response. This is
inadequate in general but will not cause trouble in our discussion here.

The Previous Action Heuristic. When any components of a system re-
sponse cannot be attributed by the above heuristics to any prior user action,
the EXPL analysis attributes them to the immediately previous user action.
This can be seen as a weakened version of the very powerful temporal suc-
cession cue in causal attribution, in which an event which follows another
immediately is likely to be seen as caused by that event (Duncker, 1945).
EXPL's encoding does not include quantitative timing information, so the
dependency of this cue on precise timing is not captured.

The previous action heuristic plays a complementary role to the obliga-
tory previous action heuristic described earlier. Obligatory previous action
ensures that the latest user action will be assigned some causal role, even if
there are no unexplained system responses. Previous action ensures that all
aspects of a system response will be assigned a cause, even if there are no
unexplained user actions.

Prerequisite Relations. In tracing the contribution of user actions to the
ultimate system response it may be necessary to recognize that an action
contributes to an intermediate system response that permits a later action to
be carried out. EXPL can make this determination in some special cases,
but the examples discussed below do not require it. The interested reader
can consult Lewis (1986a) for a description of the mechanism.

Applying the Heuristics. The heuristics are implemented by a PROLOG
program which processes the events in an example in chronological order.
Each heuristic is applied in the order listed above to each system response,
and places links between earlier user actions and components of the response.
The order of application dictates that any attributions based on identity will
be made before any based on loose-ends, for example. This order of applica-
tion is intended to ensure that links placed on the basis of definite evidence
(identity), or to satisfy a strong constraint (obligatory previous action), are
placed before the loose-ends heuristic attempts to account for unexplained
aspects of responses. In applying a heuristic the components within an event
are processed in order, which is assumed to be arbitrary.

Analysis of an Example. Figure 2 shows the output of EXPL's process-
ing of the example in Figure 1. In processing the system response “remove
train” the identity heuristic is first applied, placing the link connecting
“train” to the user action “touch train” that contains the identical compo-
nent “train.” After this link is placed, the obligatory previous action heuristic
is tried, but because there is already a link leading from the previous
action no new link need be added. The loose-ends heuristic is applied next. The component “remove” of the system response is unexplained, as is the user action “type d.” Accordingly, the loose-ends heuristic places a link connecting “remove” to “type d.” Note that EXPL’s attributions agree well with an intuitive interpretation of the English version in Figure 1.

Role of Prior Knowledge and Subsequent Experience. The EXPL heuristics assume nothing in the way of prior knowledge, other than what may be implicit in the decisions made in encoding events in a particular way. Undoubtedly, prior knowledge plays a substantial role in the analysis of real examples, when learners have some familiarity with the system and the tasks being performed.

EXPL also gives no account of the fate of analyses which are proved incorrect by later experience. A complete theory would have to describe the process by which initial conjectures, such as those developed by EXPL, are refined and revised. The complete PUPS model (Anderson & Thompson, 1986) includes a discrimination process that might be used.

Using PUPS-style Generalization on an Example Analyzed by EXPL. To support PUPS the results of EXPL’s analysis must be converted to the form assumed by the PUPS machinery, in which the procedure to be modified is explicitly represented, and the roles of its parts, when these are known, are specified. Figure 3a shows the resulting information expressed informally. The first line merely links the encoded procedure with its outcome; PUPS would use this kind of information to retrieve procedures whose outcome is similar to some desired outcome. The two role specifications that follow are just a different representation of the two links that the EXPL analyzer places when it processes the example, as shown in Figure 2.

The PUPS machinery now accepts the statement of a new outcome. It constructs a mapping to take the old outcome to the new one, in the form of a set of substitutions, as shown in Figure 3b. It then applies this mapping to the old procedure.
Outcome of $[[\text{type } d], [\text{touch train }]]$ is $[\text{remove train }]$.
Role of $[\text{type } d]$ is $[\text{specify remove }]$.
Role of $[\text{touch train }]$ is $[\text{specify train }]$.

**Figure 3a.** EXPL output as provided to PUPS for example in Figure 1.

Old outcome is $[\text{remove train }]$.

New, desired outcome is $[\text{shrink train }]$.

Substituting $\text{shrink}$ for $\text{remove}$ maps old outcome to new outcome.

**Figure 3b.** Determining mapping in PUPS.

$u \text{ type } r$
$u \text{ touch car}$
$s \text{ shrink car}$

Results of EXPL analysis of auxiliary example:

Outcome of $[[\text{type } r], [\text{touch car }]]$ is $[\text{shrink car }]$.
Role of $[\text{type } r]$ is $[\text{specify shrink }]$.
Role of $[\text{touch car }]$ is $[\text{specify car }]$.

**Figure 3c.** Auxiliary example showing shrink operation.

Original procedure:

$[[\text{type } d], [\text{touch train }]]$

Substitution does not apply to $[\text{type } d]$.

But role of $[\text{type } d]$ is $[\text{specify remove }]$.

Substitution transforms this to $[\text{specify shrink }]$.

Analysis of auxiliary example shows that $[\text{type } r]$ plays this role.$[\text{type } r]$ replaces $[\text{type } d]$.

Substitution does not apply to $[\text{touch train }]$ or its role.

Resulting modified procedure is $[[\text{type } r], [\text{touch train }]]$.

**Figure 3d.** Applying substitution of shrink for remove to the example.
If a part has no substitution, but does have a role specified, PUPS attempts to make substitutions in the role, and then to find a new part that implements the modified role. In general, background knowledge, or knowledge gleaned from other examples, will be needed here. Figure 3c shows the results of analyzing another example, part of which will be needed in modifying the current one.

The role-mapping process is shown in Figure 3d. The resulting procedure adapts the example using knowledge gathered from the auxiliary example.

**Using Synthetic Generalization on an Example Analyzed by EXPL.** Synthetic generalization requires the results of EXPL’s analysis to be cast in a different form. The links in Figure 2 are extracted from the example and combined with similar links extracted from the analysis of the example shown in Figure 3c to produce the collection of links shown in Figure 4a.

Given a new outcome, the synthetic generalizer selects from its data base of links actions which will contribute the needed components. It presumes that performing the actions linked to the desired components will produce an outcome with those components, so it simply concatenates these actions. Figure 4b shows the resulting procedure.

It is obvious that many examples would require much more sophistication than this one does. Actions could interact, or could have prerequisites. EXPL’s synthetic generalizer is a little more powerful than shown here, but not much, and many real examples would exceed its capabilities. See Lewis (1986a) for a more detailed discussion.

**Adding Substitution to Synthetic Generalization.** The example just discussed shows how synthetic generalization can combine the analysis of two examples to build a new procedure. If only one example is available EXPL’s version of synthetic generalization uses a simple substitution scheme to generalize the single example. Components are assigned to classes, as part

```
link([type d], remove)
link([touch train], train)
link([type r], shrink)
link([touch car], car)
```

**Figure 4a.** Links extracted from Figure 2 and from example in Figure 3c.

Outcome: [shrink train]

Procedure: [[type r], [touch train]]

**Figure 4b.** Procedure constructed for new outcome using links in Figure 4a.
of the encoding process, so that pictures on the screen might form one class, names of files another class, and so on. If a component is sought, but no link is available that can provide it, a search is made for identity links that provide a component of the same class. If one is found, the associated user action is modified by substituting the new component for the old one. The modified action is presumed to produce the new component. For example, if clicking on a picture of a hat is seen to be a way to specify the picture of the hat, then clicking on a picture of a fish would be presumed to be a way of specifying the picture of the fish.

This extension of synthetic generalization can be seen as the inclusion of part of the PUPS machinery, specifically the use of substitution, in the synthetic generalization framework. Without it, synthetic generalization is unable to generalize many procedures without using links derived from other examples.

**EMPIRICAL EVALUATION**

The EXPL model shows concretely how causal attribution can produce an analysis of an example which can be used by different analysis-based generalization techniques. How well does this model, or any of its variants that embodies a particular generalization method, account for human behavior in analyzing and generalizing procedures? Rather than attempting yes-or-no tests of such complex models in their entirety, I identified specific questions whose answers would be informative about individual causal attribution heuristics and about distinguishable subclasses of generalization methods.

To study these questions paper-and-pencil tasks were devised in which simple fictitious computer interactions were presented as a sequence of events in text form, with a picture showing the contents of the computer screen. Participants were asked to answer questions about the roles of particular steps in the examples, or to indicate how they would accomplish a related task. Items were constructed to probe the following issues.

*Use of Identity and Loose-ends Heuristics.* The loose-ends heuristic should permit participants to assign a role to a step by a process of elimination, even when that step contains no particular cue for what its role might be. The identity heuristic should set up the elimination process by previously linking some steps to some aspects of system responses, thus excluding them as candidate loose-ends.

*Use of Obligatory Previous Action Heuristic.* If a step with no obvious role immediately precedes a system response the obligatory previous action heuristic will assign it a role, whereas the same step appearing in the midst of a sequence of user actions might not be assigned any role.
Rationalistic versus Superstitious Generalization. As discussed above, superstitious generalization will normally preserve order of steps, while rationalistic generalization will accept reorderings as long as no logical constraint, such a prerequisite relationship between two steps, is violated. An example was constructed in which two steps could be reordered without violating any apparent constraint, and participants were asked to judge whether the reordered example would work.

Another item examined the treatment of an uninterpreted step. As discussed earlier a superstitious generalizer will leave unchanged aspects of the example to which it has assigned no role, since it has no basis for modifying them. A rationalistic generalizer will show the opposite handling: only interpreted steps can appear in a generalization, since steps will be included in a procedure only if they contribute to the goal for which the procedure is being built. An example was prepared that included an apparently unnecessary step. While some participants might assign a role to the step, it is possible that participants who assigned it no role would nevertheless keep it in a generalization.

Generalizing About Novel Material. EBG and EBL rely on prior domain theories to generalize examples. As discussed earlier, it seems likely that people can understand and generalize about procedures for which they lack such a theory. All items included meaningless tokens to which EXPL could assign a role and then generalize about. Participants' handling of these materials should indicate whether they are or are not dependent on a prior domain theory.

In constructing the items, I exploited a convenient feature of the human-computer interaction domain: People know that computer commands contain a mixture of meaningless and meaningful material. Tolerance of meaningless commands makes it possible to use the same command in different contexts so as to see whether the context, rather than the command itself, determines the interpretation learners place on it. The loose-ends heuristic should produce this kind of context effect. Tolerance of meaningful commands, on the other hand, permits some control over probable interpretations of parts of examples where this is desired. In probing the handling of unnecessary steps it is useful to present them in a context in which other steps have a natural interpretation which is adequate to explain the observed outcome. Use of these techniques results in test items which contain both meaningful and meaningless material.

Method

Participants. Ninety students in an introductory psychology course served in the experiment as part of a course requirement. As a rough gauge of computer background they were asked to estimate hours of computer use. Esti-
mating ranged from 0 to 1000, with a median of 55 and lower and upper quartiles of 20 and 100.

**Materials.** Test items were presented on single pages of test booklets. Each page carried the name of a fictional computer system, with a sketch of a display screen and (if used in the example) a keyboard. A brief example of an interaction with the system was then presented as a sequence of written steps, followed by one or more questions about the example. Figure 5 shows the picture for a typical item; the example and question were placed on the same page immediately below the picture. Table 1 shows the content of each item. Because the logic underlying the construction of the individual items differs considerably, I have incorporated the detailed discussion of each item with the presentation of results, below. Groups of participants were given different versions of the booklets, differing in the items included and the order of certain items, as shown in Table 2. Items TRAIN, PERSON, and HOUSE relate to the problem of identifying hidden events in analyzing procedures and will not be discussed here.

All booklets contained an initial practice item, which was discussed with participants at the start of the experimental session, and a final page with background questions on computer use.

**Procedure.** Participants were run in groups of five to twenty in a classroom. In early sessions participants were assigned to Groups A and B in alternation on arrival; later Groups S and T were formed in the same manner. Participants were given instructions verbally. Points covered were that
<table>
<thead>
<tr>
<th>Item</th>
<th>In Picture</th>
<th>Example</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUCK</td>
<td>truck and boat</td>
<td>1. Type &quot;67m&quot; on keyboard.</td>
<td>What does Step 1 do?</td>
</tr>
<tr>
<td>Form 1</td>
<td>on screen, keyboard</td>
<td>2. Type &quot;truck on keyboard.&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ditto</td>
<td>&gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; Truck turns red.</td>
<td>ditto</td>
</tr>
<tr>
<td>Form 2</td>
<td></td>
<td>1. Type &quot;67m&quot; on keyboard.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Type &quot;red&quot; on keyboard.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; Truck turns red.</td>
<td></td>
</tr>
<tr>
<td>LADDER</td>
<td>tree and ladder</td>
<td>1. Type &quot;NNA&quot; on keyboard.</td>
<td>What would you do to make the ladder shrink?</td>
</tr>
<tr>
<td></td>
<td>on screen, keyboard</td>
<td>2. Type &quot;ladder&quot; on keyboard.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; Ladder rotates 45°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Type &quot;NNA&quot; on keyboard.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4. Type &quot;da9&quot; on keyboard.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; Tree rotates 45°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5. Type &quot;n6b&quot; on keyboard.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; Tree shrinks to half size.</td>
<td></td>
</tr>
<tr>
<td>MANAGERS</td>
<td>blank screen</td>
<td>1. Type &quot;display3&quot;.</td>
<td>Which step would you change if you wanted a</td>
</tr>
<tr>
<td>Form 1</td>
<td>keyboard</td>
<td>2. Type &quot;n25&quot;.</td>
<td>list of managers' ages instead of managers'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; &gt;&gt; System shows list of manager's salaries.</td>
<td>salaries?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Which step would you change if you wanted a</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>list of clerks' salaries instead of managers'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>salaries?</td>
</tr>
</tbody>
</table>

continued
<table>
<thead>
<tr>
<th>Item</th>
<th>In Picture</th>
<th>Example</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>MANAGERS (cont.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Form 2</td>
<td></td>
<td>1. Type &quot;n25&quot;.</td>
<td>ditto</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Type &quot;display3&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; &gt; &gt; &gt; &gt; &gt; System shows list of managers' salaries.</td>
<td></td>
</tr>
<tr>
<td>STAR</td>
<td></td>
<td>1. Touch the star.</td>
<td>If I tried to move the star to the bottom of the screen this way:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Touch &quot;beta&quot;.</td>
<td>Touch &quot;beta&quot;.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Touch a place near the left side of the screen.</td>
<td>Touch the star.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; &gt; &gt; &gt; &gt; &gt; The star moves to the left side of the screen.</td>
<td>Touch a place near the bottom of the screen.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Would it work? If not, why not?</td>
</tr>
<tr>
<td>FISH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Type &quot;delete&quot; on the keyboard.</td>
<td>What does Step 2 do?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Type &quot;c43&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Type &quot;hat&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; &gt; &gt; &gt; &gt; &gt; The hat disappears.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RABBIT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Type &quot;rabbit&quot;.</td>
<td>What does Step 3 do?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Type &quot;remove&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Type &quot;HJ4&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; &gt; &gt; &gt; &gt; &gt; Rabbit disappears.</td>
<td></td>
</tr>
</tbody>
</table>
questions were intended to investigate their interpretations of the examples, regardless of the amount of their knowledge of computers, that each item referred to a different fictitious computer system, that accordingly they should not attempt to correlate their answers to different items or go back and change earlier answers. The use of a touch screen, in examples where no keyboard was used, was explained. Participants were asked to look at the practice item and to suggest possible roles for its first step. It was stressed that there were no correct or incorrect answers since the intent was to discover each person's interpretation of the examples, and that participants were free to indicate when they could not determine an answer. Participants were then asked to begin work, moving at their own pace, and to turn in their booklets and leave when finished.

Coding and Analysis of Responses. Coding categories, shown in the table of results for each item, were constructed for each item before any responses were examined. Three raters coded all responses independently, with final codes assigned by majority rule. Responses for which no two raters agreed were coded as “no agreement.” No codes were discussed among the raters, either during the rating process or in the assignment of final codes. The G or log likelihood ratio test (Sokal & Rohlf, 1981) was used to test for differences in response frequencies.

Results
Table 3 shows the responses for each item. Where the same item was presented to more than one group, G tests did not indicate significant intergroup differences, except in the case of item RABBIT. Accordingly, results are pooled across groups except in that case.
<table>
<thead>
<tr>
<th>Item</th>
<th>Number of Responses</th>
<th>Category of Response</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Form 1</td>
<td>Form 2</td>
</tr>
<tr>
<td>TRUCK</td>
<td>0</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>LADDER</td>
<td>70</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>MANAGERS</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>First Question</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Second Question</td>
<td>25</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>STAR</td>
<td>19</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>53</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>FISH</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>First Question</td>
<td>26</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>53</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Second Question</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>57</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>RABBIT</td>
<td>Group S</td>
<td>Group T</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 4
Interpretation of Step in Item TRUCK

<table>
<thead>
<tr>
<th>Content of Step 2</th>
<th>Interpretation of Step 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{truck}</td>
<td>30 31</td>
</tr>
<tr>
<td>\textbf{red}</td>
<td>1 31</td>
</tr>
</tbody>
</table>

\[ G = 61, 1 \text{ df}, p < .001 \]

**Item TRUCK.** This item was given in two forms, one with the second step containing "truck," the other with the second step containing "red." Together, the identity and loose-ends heuristics should result in the first step, which is the same in both items, being assigned the role of specifying the aspect of the system response that is not mentioned in the second step.

This is confirmed by the data. Table 4 tabulates just those responses indicating a specification of color or of object or location. The difference due to the form of the item is highly significant \( G = 61, 1 \text{ df}, p < .001 \).

**Item LADDER.** This item examines whether attributions made using identity and loose-ends in an earlier part of an example can be carried forward to disambiguate later phases of an example. Identity and loose-ends should indicate that "NNA" specifies rotation in analyzing steps 1 and 2. If this interpretation is carried forward to steps 3 and 4 the analysis will indicate that "da9" specifies the tree. Finally, analysis of steps 5 and 6 will connect "n6b" with shrink, given the connection of "da9" with tree.

Most participants responded in a manner consistent with this analysis, but there are other possible explanations of this result. It is possible that participants assume that items always consist of an operation followed by an operand, and associate "n6b" with "shrink" on this basis.

**Item MANAGERS.** This item provides a test of the interaction of the loose-ends heuristic, the previous action heuristic, and the obligatory previous action heuristic. Assume that the steps in the examples are encoded as shown in Figure 6a: typing the meaningful term "display" is separated from typing "3". Assume further that the relationship between "display" and "show list of" is known and available to establish an identity link accounting for this aspect of the system response. Figure 6b shows the state of analysis following construction of this identity link. Note that in neither form is there a link drawn from the last user action to any later system response. If the obligatory previous action heuristic is now applied, as in the EXPL implementation, a link will be placed attributing the first unaccounted-
Figure 6a. Encoding of Forms 1 and 2 of MANAGERS item.

Figure 6b. After placement of identity links.

Figure 6c. After applying obligatory previous action heuristic.

Figure 6d. After applying loose-ends heuristic.

Figure 6e. Result of eliminating obligatory previous action heuristic.
for component of the system response to the previous action, as shown in Figure 6c. The loose-ends heuristic will now connect any unattributed components of the system response to the earliest unaccounted-for user action, with results shown in Figure 6d. This analysis predicts that participants seeing Form 1 would attribute "manager's" to step 2 and "salaries" to step 1, while participants seeing Form 2 should attribute "manager's" to step 1 and "salaries" to step 2. As the tabulation in Table 5 shows, this pattern does not occur.

If the obligatory previous action heuristic is not used the analyses obtained are shown in Figure 6e. As can be seen, the attributions are consistent with the dominant pattern of participants' responses.

Although a modified EXPL analysis can account for these results it seems imprudent to attach much weight to these examples in assessing the interactions of the heuristics. The items have the drawback that the analysis is heavily dependent on encoding, including the order of components. A change in encoding of the system response from "show manager salary" to "show salary manager," for example, would change EXPL's analysis.

In view of the uncertainty in EXPL's treatment it is interesting that participants were so consistent in their attributions in these impoverished examples. Possibly, participants were influenced strongly by the order in which the questions were asked, attributing the first effect they were asked about to the most recent step, and then choosing not to attribute two effects to the same step.

**Item STAR.** Most participants indicate that the reordered procedure will not work, without giving a reason beyond the change in order. As discussed earlier, this would be expected from a superstitious generalization process. On the other hand, 19 participants indicate that the reordered procedure would work, consistent with rationalistic generalization. The 95% confidence interval for proportion of participants accepting the change of order, ignoring uninterpretable responses, extends from .07 to .46.

While retention of order is expected under superstitious generalization, it is not completely inconsistent with rationalistic generalization. Participants might have a belief that order of steps is generally important in computer

<table>
<thead>
<tr>
<th>Answers to Questions</th>
<th>Form 1</th>
<th>Form 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>step 1, step 1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>step 1, step 2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>step 2, step 1</td>
<td>23</td>
<td>20</td>
</tr>
<tr>
<td>step 2, step 2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
procedures, and could apply that belief in either of two ways. First, they
could incorporate specific order constraints on the example’s operations
into their analysis (EXPL’s synthetic generalizer uses a very simple planner
that cannot handle arbitrary order constraints, but a synthetic generalizer
with a more sophisticated planner could). Alternatively, they could use
order in criticizing a proposed procedure, even if their generalizer is ration-
alistic. Since the test item presented the original procedure along with a pro-
posed variant such criticism would have been easy.

A belief that order of steps is important in computer procedures might be
learned. Table 6 tallies acceptance of variant order and rejection of variant
order with no grounds for participants reporting less and more than the
median computer experience. As can be seen there is no indication that
more experienced participants are less likely to accept the variant order.
This lack of dependence on experience is not decisive, however: even people
with no or very little experience might have the impression that computer
procedures are inflexible, and people with more experience might have more
exposure to systems that are actually more flexible.

Item FISH. As discussed above, superstitious and rationalistic general-
ization differ in their treatment of uninterpreted steps. Table 7 tabulates
participants according to whether they assigned a role to the seemingly un-
necessary Step 2, and whether they retained this step in generalizing the ex-
ample. As can be seen, 23 participants retained the step even though they
assigned no role to it, consistent with a superstitious generalization mecha-
nism but not consistent with rationalistic generalization. On the other hand,
7 participants dropped the uninterpreted step, which is consistent only with
rationalistic generalization. One participant neatly combined rationalistic
with superstitious generalization by suggesting that Step 2 be dropped, but
put back in if the new procedure did not work without it.

When participants assigned roles to ‘c43’ they treated it appropriately in
the generalized procedure, consistent with all of the generalization models

<table>
<thead>
<tr>
<th>TABLE 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationship of Acceptance of Variant Order in Item STAR with Experience</td>
</tr>
<tr>
<td>Response to new order in STAR</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>will work</td>
</tr>
<tr>
<td>order bad, no reason given</td>
</tr>
</tbody>
</table>
considered here. Typical roles included indicating the position of the hat, specifying a location in memory for the hat to be put, requesting that Step 1 should be executed, and indicating that the next object touched should be acted upon. The lone participant who dropped ‘c43’ from the generalized procedure after giving it a role said that it caused the system to exclude the fish from the deletion operation.

Table 8 compares responses to the FISH item with those of the STAR item. If use of rationalistic or superstitious generalization were consistent by participant, participants should fall mainly in the “will work, drop” cell, for rationalistic, or the “order bad, keep” cell, for superstitious generalization. To the contrary, more participants fall in the other two cells, suggesting inconsistency across the two items. The “will work, drop” cell is empty, suggesting that no participants were consistently rationalistic, while some were consistently superstitious and others were superstitious on one example and not the other.

Item FISH illuminates another point discussed above. Most participants generalized the example by replacing Hat by Fish, even though they had seen no example in which Fish was typed. This generalization is trivial in PUPS but cannot be handled in synthetic generalization without adding substitution.
**Item RABBIT.** This item showed a significant effect of order, so results are not pooled across groups. The comparison between this item and FISH provides a test of the obligatory previous action heuristic. According to this heuristic even an apparently unnecessary step must be assigned a role if it immediately precedes a system response. In FISH the unnecessary step occurs between two user actions, while in RABBIT it occurs just before a system response. As shown in Table 9 there is some support for the obligatory previous action idea in that of the participants who assigned a role in one and not the other nearly all assigned a role in RABBIT and not in FISH. This preponderance is significant by sign test at the 95% level in each group. But the table also shows that most participants assigned a role to the unnecessary step in both examples. This indicates that analysis should attempt to assign a role to all actions, regardless of position, rather than giving special handling to actions that immediately precede a system response. This finding joins the results of the MANAGERS item in casting doubt on EXPL's obligatory previous action heuristic.

**Discussion**

**Support for Analysis Heuristics.** The empirical findings support the conclusion that people use principles similar to EXPL's identity and loose-ends heuristics. The detailed coordination of these heuristics is less clear, and may differ from that in the implemented EXPL system. It appears that people tend to assign a role to all user actions, regardless of position, rather than using EXPL's obligatory previous action heuristic.

**Superstition or Rationalism?** As noted above, items STAR and FISH produced a preponderance of responses suggestive of superstitious generalization, but many participants were apparently inconsistent across the two

### TABLE 9

<table>
<thead>
<tr>
<th>Interpretation of 'c43' in FISH</th>
<th>Interpretation of 'HJ4' in RABBIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>given role</td>
<td>Group S</td>
</tr>
<tr>
<td>no role or role not known</td>
<td>21</td>
</tr>
<tr>
<td>given role</td>
<td>9</td>
</tr>
</tbody>
</table>
items. Further, responses on item STAR could have been influenced by specific beliefs about the importance of order in computer procedures. What can be concluded from the data? Let us consider some possible interpretations, focussing on the 26 participants who provided relevant responses to items STAR and FISH, tallied in Table 8.

All participants rationalistic: All participants consistently use rationalistic generalization. Apparently superstitious responses are produced by the influence of background beliefs about computers. This would account for the occurrence of superstitious responses on item STAR, but fails to account for superstitious responses on item FISH. Rationalistic generalization cannot produce procedures with unexplained parts, so this interpretation would require that participants had explanations for the mystery step in the procedure which they could not or would not express. Discounting this possibility, at most the 7 participants who dropped the mystery step in FISH can be consistently rationalistic.

All participants superstitious: All participants consistently use superstitious generalization, with apparently rationalistic responses produced by some other mechanism. The difficulty here is seeing what this other mechanism could be. Superstitious generalization cannot omit unexplained steps. Item STAR might provide more leeway, since participants did not actually generate the variant order procedure but only had to accept or reject it, but it is hard to see how a superstitious generalizer could establish the acceptability of a reordered procedure. It appears that at most the 12 participants who rejected the modified order in item STAR and retained the mystery step in FISH can be consistently superstitious.

Consistent but mixed: All participants are consistent, but some are superstitious and some rationalistic. As argued above, at most 7 of the participants can be consistently rationalistic, even allowing for seemingly superstitious responses to STAR. But the remaining 19 participants cannot all be consistently superstitious: as just argued, at most 12 of them are.

Some participants inconsistent: It appears that at least the 7 participants who accepted the variant order for STAR, but retained the mystery step in FISH, are using rationalistic generalization on STAR and superstitious generalization on FISH. The 12 participants who rejected the reordering in STAR and retained the mystery step in FISH are behaving superstitiously on FISH and may or may not be rationalistic on STAR, allowing for the role of background beliefs about order. The 7 participants who accepted the variant order for STAR but dropped the mystery step in FISH
istically in FISH and may or may not have generalized rationalistically in STAR. We can conclude that at least 7 of the participants were inconsistent, and it is possible that all were.

Looking at item FISH alone, including participants who did not provide relevant responses for STAR, we see in Table 7 that 7 participants used rationalistic generalization and 23 were superstitious. So, for those participants providing interpretable data on this item a clear preponderance were superstitious. But the participants who assigned a role to the mystery step, and hence did not provide diagnostic data, may have been largely rationalistic generalizers. Therefore, we cannot conclude that most participants were superstitious on this item, even though this is true for the participants we can classify.

A preponderance of participants gave responses for item STAR that are apparently superstitious. But, if we allow that some or all of these responses might actually come from rationalistic generalizers this does not help to determine how common the two modes of generalization are.

To summarize, a conservative interpretation of the data from both items indicates that at least some participants were inconsistent, using superstitious generalization on one item and rationalistic on another. We cannot determine whether rationalistic or superstitious generalization is more common overall. A less conservative interpretation, which discounts the possibility that background beliefs could mask rationalistic generalization in STAR, and assumes that the participants who gave relevant responses to both items are representative, indicates that superstitious generalization is more common than rationalistic generalization overall, but that many participants are inconsistent.

Whatever conclusion we accept about the prevalence of the two generalization modes, the results may well be influenced by the fact that participants had full access to the examples while interpreting or generalizing them. In real learning situations participants would usually face a serious retention problem, in which recalling complete examples well enough to use superstitious generalization might be difficult. Under these conditions rationalistic methods, which could work with even fragmentary recall of examples, might be more prevalent.

*Generalizing About Novel Material.* All items included meaningless tokens or figures for which participants could not have a prior domain theory. Yet participants were well able to generalize from these procedures. Procedures in item LADDER, for example, contained the nonsense terms NNA, da9, and n6b, but the great majority of participants assigned roles to these in a consistent way, and generated a procedure which used n6b. If participants use a domain theory they must be able to extend it to incorporate the behavior of entities they see for the first time in a to-be-generalized example.
GENERAL DISCUSSION

Generality of Causal Attribution Heuristics
To what extent are the analysis and generalization mechanisms we have been discussing dependent on knowledge of the specific domain we have considered, human-computer interaction? Could these same mechanisms be applied to concepts outside this domain, or are they embodiments of particular assumptions learners make about this particular domain, assumptions which must be the result of some prior, possibly more basic learning process?

The generalization mechanisms are clearly not limited to this domain, since they have all (except for synthetic generalization) been developed to deal with other kinds of concepts. What about the analysis heuristics?

The obligatory previous action heuristic (which did not receive strong support) is an example of a piece of machinery which might rest on special assumptions. The rationale for it as discussed above, the assumption that system responses are fast compared with user actions, certainly would not apply to all procedural domains. But this argument is not decisive, because this may not be the correct rationale. As also discussed above, temporal succession is a very powerful cue for causal attribution in domains unrelated to human-computer interaction; the obligatory previous action heuristic could reflect the tendency to attribute effects to immediately prior events, just as the plain previous action heuristic does.

The identity heuristic does not appear to rest on any specific ideas about human-computer interaction. As noted above it may be related to the similarity principle that figures in causal attribution in other domains (Shultz & Ravinsky, 1977); in the specific form used here it has been used by Anderson (1987) to analyze procedures in algebra. Nevertheless, it may reflect assumptions that are not completely general. While principles akin to identity may be involved in unravelling many physical phenomena, for example the notion that objects in the same place are more likely to interact than objects in different places, identity might seem especially useful in understanding artifacts rather than natural systems. If red and green switches are available to control red and green lights, it seems compelling that a well-meaning artificer would have matched up the colors. There seems much less warrant for the conjecture that drinking a naturally-occurring red plant extract (say) will be effective in making one's face flush red.

But of course just such conjectures are commonplace in prescientific thought; see discussion in Frazer (1964). So whatever we may think of the support for it, it appears that the identity heuristic is not restricted to artifacts, let alone computers.

The rationale proposed above for the loose-ends heuristic, like that for the obligatory previous action heuristic, would restrict its application. It was assumed that the events the learner is seeing constitute a coherent and efficient demonstration, without wasted motion or mistakes. There is nothing
in that assumption that is limited to the human-computer interaction domain specifically: One might make this assumption about a sample solution to a physics problem. But some demonstrations do contain mistakes, and many naturally-occurring procedures (procedures not intended as demonstrations) do contain steps which do not contribute to the goal; for example, procedures produced by novices. It is possible, however, that learners will apply loose-ends without making this assumption. Just as people do not restrict the use of identity to artifacts, they may tie up loose-ends when there are no grounds for expecting them to connect.

One special aspect of causal attribution in the human-computer interaction domain may be reflected in a heuristic that is not included in EXPL. Pazzani (1987) reports evidence that the mechanism principle (Bullock, Gelman, & Baillargeon, 1982), by which causal attributions are more plausible when there is a possible mechanism that could mediate the causal connection between two events, plays an important part in analyzing examples involving human actions. But in the human-computer interaction domain a mechanism is always available: the computer itself. People seem ready to accept arbitrary connections between user actions and the computer’s responses, as if the mechanism requirement is satisfied by default.

Extending Explanation-based Approaches to Deal with Novel Material
The ability of participants to generalize examples that contain arbitrary, never-seen-before tokens, as in item LADDER and others, bears out the contention that EBG and EBL, as they stand, cannot provide a complete account of learning in this domain. To attack this problem the EBG or EBL framework might be extended to include additions to the domain theory as part of the analysis of an example. The EXPL analysis machinery, for example, could be adapted to produce its output in the form of statements expressed in logic about the significance of the steps in the example, rather than as links or role assignments as needed by synthetic generalization or PUPS. The generalization process itself would work just as it does in normal EBG or EBL, but of course the results would no longer be rigorously justifiable, being only as good as the heuristically-conjectured domain theory.

Explanation-based Approaches Can Mimic Analogical and Synthetic Generalization
How would such an extended explanation-based model compare with structure mapping, PUPS or synthetic generalization? Would it be rationalistic or superstitious? The behavior depends on the nature of the domain theory. With appropriate domain theories EBG or EBL can mimic the generalizations of any of these models.
Suppose first that the domain theory specifies how the parts of a procedure produce its outcome. In this case the explanation-based model implements structure mapping. Kedar-Cabelli (1985) describes a procedure called “purpose-directed analogy” in an EBG framework. If applied to generalization of procedures purpose-directed analogy would construct new procedures by capturing the relationship between procedure and outcome in the example in the form of a proof that the procedure produces the outcome. The proof would then be generalized. The new procedure would be determined by the constraint that the generalized proof must establish that the new procedure produces the desired new outcome. This is the structure mapping process, in which the analogy \( P : O :: X : O' \) is solved by mapping the relationships in the \( P-O \) structure onto the \( X-O' \) structure.

Seen in the explanation-based framework, synthetic generalization appears as a special case of structure mapping. While structure mapping can incorporate arbitrary relationships among attributes of procedures and their outcomes, synthetic generalization requires that only general principles of combination of steps, which are implicit in the synthesis process, and specific descriptions of parts, which are produced in the analysis of an example, are permitted. Thus, the domain theory for synthetic generalization consists of two distinct subtheories. An a priori subtheory describes how parts of procedures interact when put together. This theory must be general, not referring to features of any particular examples. The second subtheory consists of descriptions of the various possible parts of procedures, whose behavior may have been extracted from the analysis of examples.

Figures 7a and b show how Item FISH could be handled in an EBG version of synthetic generalization. The a priori domain subtheory is an explicit statement of the assumption underlying EXPL’s synthetic generalization planner, without the substitution scheme. The part-specific subtheory contains relationships posited by the analyzer in processing examples. As required for pure synthetic generalization, two examples are processed, one to establish how to specify Delete and one how to specify Fish. To build a procedure for Removing Fish we take the intersection of the two goal concepts. As expected from a rationalistic approach the step c43 is dropped. The EBG machinery is doing two things here. First, it is filtering the attributes of the examples so that only apparently necessary attributes are kept. Second, it is streamlining the application of the domain theory by replacing more abstract specifications of goal concepts by more concrete ones.

It might appear that a superstitious generalization mechanism like PUPS could not be accommodated in the EBG or EBL framework. After all, one of the functions served by proofs in EBG, or schema-matching in
A priori domain theory:

A is an aspect of the outcome of procedure P if S is a step of P and S is linked to A.

Example 1:

u type delete
u type c43
u type hat
s remove hat

Assertions added to domain theory by analysis of Example 1:

[type delete] is linked to remove.
[type hat] is linked to hat.

Note that [type c43] has been given no role.

Example 2:

u type reduce
u type fish
s shrink fish

Assertions added to domain theory by analysis of Example 2:

[type reduce] is linked to shrink.
[type fish] is linked to fish.

Figure 7a. Using EBG to perform synthetic generalization for Item FISH.

EBL, is filtering out features of examples that have no role, while PUPS simply retains features without roles. Nevertheless, with an appropriate domain theory the explanation-based mechanisms can mimic PUPS, at least in simple cases.

The domain theory needed for PUPS is somewhat different from those for structure mapping or synthetic generalization. While theories for those models will describe the role of all relevant parts of procedures, the theory for PUPS may not. Instead, the PUPS theory must indicate the outcome of a procedure as a whole, so that uninterpreted parts will not be stripped out in the generalization process. To permit generalization to work at all on the whole procedure, any replaceable parts must be detected and replaced by variables in the domain theory. Thus, the domain theory represents a procedure as a sort of matrix in which some parts, those with known roles, are substitutable, while parts without known roles are fixed.

Figures 8a and 8b show the treatment of Item FISH in a PUPS-like version of EBG. Note that the analysis of the example must perform a good deal of abstraction, but that the relationships that must be detected to do this are the same as are needed for synthetic generalization, and are detected
Goal Concept 1:
Procedure-outcome pairs (P,O) such that remove is an aspect of the outcome of P.

Proof that Example 1 is a member of Goal Concept 1:
[type delete] is a step of Example 1.
[type delete] is linked to remove.
Therefore remove is an aspect of the outcome of Example 1.

Generalization based on proof:
(P,O) is in Goal Concept 1 if [type delete] is a step of P.

Goal Concept 2:
Procedure-outcome pairs (P,O) such that fish is an aspect of the outcome of P.

Proof that Example 2 is a member of Goal Concept 2:
[type fish] is a step of Example 2.
[type fish] is linked to fish.
Therefore fish is an aspect of the outcome of Example 2.

Generalization based on proof:
(P,O) is in Goal Concept 2 if [type fish] is a step of P.

Construction of procedure to accomplish [remove fish]:
Desired procedure P is in a pair that lies in the intersection of Goal Concepts 1 and 2. If [type delete] is a step of P, and [type fish] is a step of P, (P,O) will be in Goal Concepts 1 and 2. Note that [type cat] is not included in the construction.

Figure 7b. Continuation of Figure 7a.

by EXPL’s analyzer: the step [type delete] specifies remove, the step [type hat] specifies hat. The required abstractions are accomplished by replacing tokens that appear in both the procedure (or roles of parts of the procedure) and the outcome by variables.

Explicit and Implicit Generalization. This argument that EBG or EBL can mimic other generalization methods does not mean that EBG and EBL cannot be distinguished from the other methods. Recall that EBG and EBL produce explicit generalizations, while the other methods do not. It might be possible to determine whether human learners produce explicit generalizations from examples, or whether, as in the other methods, generalization happens implicitly in response to the demand to accomplish a new outcome. This is likely to be difficult. Synthetic generalization, for example, while it does not produce explicit generalizations, does reduce examples to an ab-
Example:

u type delete
u type c43
u type hat
s remove hat

Domain theory constructed from example:

(1) Outcome of [X, [type c43], Y] is [Q R] if

role of X is [specify Q] and

role of Y is [specify R].

(2) Role of [type delete] is [specify remove].

(3) Role of [type Z] is [specify Z].

Goal concept:

Pairs P,O such that the outcome of procedure P is O.

Figure 8a. PUPS-like generalization in EBG.

Abstract form which it then uses in building new procedures. Discriminating such an abstract representation of an example from a generalization appears hard.

Discriminating and EBG implementation of PUPS from real PUPS also looks difficult. EBG-PUPS builds a domain theory from an example, and discards the example itself, before processing a new goal. One might expect, therefore, that comparing people's ability to recall examples with their ability to generalize from them, at varying delays, would reveal that people could generalize from examples they could not recall, if EBG-PUPS were used. But this overlooks the fact that the domain theory constructed from an example in EBG-PUPS actually contains all the information in the original example, including arbitrary order and unexplained steps.

How Well Does Analysis-based Generalization Fit the Data?

Clearly, participants were well able to produce generalizations, consistent between participants, from single examples. This expected finding confirms
Proof that the example and its outcome satisfy the goal concept:

Let \( X = [\text{type delete}] \)
\[ Y = [\text{type hat}] \]
\[ Q = \text{remove} \]
\[ R = \text{hat} \]
\[ W = \text{hat}. \]

Role of \([\text{type delete}]\) is \([\text{specify remove}]\) by assertion (2) in domain theory, so role of \(X\) is \([\text{specify } Q]\).

Role of \([\text{type W}]\) is \([\text{specify } W]\) by assertion (3) in domain theory, so role of \([\text{type hat}]\) is \([\text{specify hat}]\) and therefore role of \(Y\) is \([\text{specify } R]\).

Since the conditions on \(X, Q, Y,\) and \(S\) in (1) are satisfied, the outcome of \([X, [\text{type c43}], Y]\) is \([Q R]\); that is, the outcome of \(([\text{type delete}], [\text{type c43}], [\text{type hat}])\) is \([\text{remove hat}]\).

Generalization based on proof:

Replacing \(\text{hat}\) by a variable, and leaving other terms in the example fixed, we find that any procedure

\([\text{type delete}], [\text{type c43}], [\text{type Z}]\]
and outcome
\([\text{remove Z}]\)

are in the goal concept.

Therefore to get \([\text{remove fish}]\) use \([\text{type delete}], [\text{type c43}], [\text{type fish}]\).

**Figure 8b.** Continuation of Figure 8a.

that inductive, or similarity-based, generalization methods do not provide a good account of behavior in this domain. While the details of EXPL's causal attribution procedure did not receive much support, the data suggest that the identity and loose-ends heuristics play a role in people's analysis of procedures. Participants were able to generalize about material for which they lacked a prior domain theory, so explanation-based methods that rely on a domain theory must be extended to permit the domain theory to be extended during the analysis of an example. If this is done, explanation-based methods become able to mimic any of the other methods considered here, including both superstitious and rationalistic ones.

The items contrasting superstitious and rationalistic generalization revealed at least some inconsistency in individual participants, indicating that no single superstitious or rationalistic method can account for all of the data. Explanation-based methods could account for this inconsistency, since
they provide a single framework in which both kinds of generalization can be accomplished. An alternative, but more complex, possibility is that participants actually employ two different generalization methods, a rationalistic one, such as synthetic generalization, and a superstitious one, such as PUPS.

Why would people shift between different styles of domain theory in an explanation-based framework, or between two different generalization methods? In the conditions of this study, when to-be-generalized material was always available for examination, and where-to-be-accomplished outcomes were quite simply related to the outcomes of the examples, it is hard to see why anything other than superstitious generalization would be used. Indeed, the data suggest, though they do not prove, that superstitious generalization was the commoner mode in the study. In other circumstances, however, participants would have trouble retaining the details of examples, and would have to attempt goals more remote from those achieved in examples they have seen. Rationalistic methods might be more effective in these cases. It is easy to see, therefore, why people might maintain a mixed repertoire of generalization methods, but it is less clear why some participants employed mixed methods in the study.

Back to Wertheimer: Analysis and Understanding
What is the relationship between the analyses that are used by these generalization methods and the notion of understanding developed by Wertheimer? It appears that in each method the analysis captures part of Wertheimer's idea, and provides a concrete working-out of what understanding can contribute to transfer, but also falls short of representing everything Wertheimer argues is critical in understanding.

Wertheimer and EBG. The relationship between EBG and Wertheimer's conception is easiest to articulate, because Wertheimer dealt explicitly with the difference between proof and understanding. In EBG, the important relations among parts of an example are picked out by their use in a proof that the example is a member of the to-be-learned concept. The identification of "important relation" with "relation that figures in a proof" is attractive, but Wertheimer rejects it. In his discussion of the parallelogram problem he gives a proof of the formula for the area of a rectangle which he argues does not reflect understanding, because it does not incorporate what he sees as the critical geometric insight in the problem, that the area of a rectangle is made up of the sums of the areas of so many rows or columns. Thus, while any proof is acceptable as an analysis in EBG, only some proofs are acceptable to Wertheimer.

The Problem of Choosing an Analysis. What is the significance of Wertheimer's qualification when applied in the EBG framework? It calls atten-
tion to the fact that different proofs of the same proposition are possible, and that they will support different generalizations. In his example, Wertheimer contrasts the "sum of rows or columns" proof for the area of a rectangle with a proof which derives the area of a rectangle algebraically from the formula for the area of a square, using a construction which decomposes a large square into smaller squares and rectangles. The former proof generalizes to a variety of figures to which the second does not, for example, a stair-step parallelogram formed by sliding some rows of a rectangle to the right. Thus, the issue of choosing or even designing a proof so that it will support desired generalizations is raised.

What guidance does Wertheimer offer for selecting the right analysis? For Wertheimer, good analyses are characterized by their internal "fit." One must search a space of representations, which differ in the grouping of situation elements and the choice of central elements, looking for "structural truth" (Wertheimer, 1959, pp. 234-236.) If one connects this conception with the problem of generalization the implicit argument is that representations with good fit will support useful generalizations, while representations with poor fit will support trivial generalizations.

It seems unlikely that this internal criterion for analyses can be sufficient. Greeno (1978, 1983) extends Wertheimer's framework to include interconnections with other knowledge as a criterion for quality of understanding. For example, good understanding of the fact that multiplication and division are inverses requires recognizing the connection between this pair of inverses and a broad class of other pairs, such as freezing and thawing. Riley (1986) discusses the application of Greeno's ideas to human-computer interaction.

It is plausible that the external connections Greeno calls for could help in generalizing an analysis to other situations. But Wertheimer's examples can be used to argue that the notion of graded understanding, in which there are better and worse ways to think about a problem, needs to be replaced by a more differentiated view in which different analyses simply support different generalizations. Thus, analyses with better or poorer internal fit, or more or fewer external connections, may on the average support more or fewer generalizations, but analyses that look roughly the same on these dimensions will differ in just which generalizations they support.

**Choice of Analysis Depends on Specific Generalization Goals.** Consider the problem of determining the area of a parallelogram. Wertheimer favors an analysis in which the part of the parallelogram that juts out on one end is seen to fit in a gap on the other end. Rearranging the parts forms a rectangle, whose area (it is assumed) can be found. This analysis generalizes to a wide variety of figures, such as a jigsaw puzzle shape in which a protruding ear on one side can fill a socket on another side, as shown in Figure 9a. A dif-
different analysis views the parallelogram as a rectangle whose (very thin) rows have been slid over, so that the figure slopes up on one side and overhangs on the other. Since sliding does not change the area of a row the area of the whole figure is unchanged. This analysis will not apply to the jigsaw puzzle form, because that form cannot be produced just by sliding rows. But the sliding analysis will apply to some forms to which the gap-filling analysis will not. Consider a chevron formed from a tall, narrow rectangle by sliding the center of the rectangle far to the right, as shown in Figure 9b. Cutting off the right-hand portion of the resulting form, and replacing it on the left, will not produce a rectangle. Many such cuts and replacements would be needed before a rectangle would result. In fact it is not easy to see that a rectangle would ever result without insight from the sliding analysis.

No matter how Wertheimer might evaluate the internal fit of this second analysis, the analyses cannot be ordered in generality in any simple way. Each can be extended in ways that the other cannot. A system seeking generalizations must choose the right analysis for the problems it will face, or must provide itself with both.

These examples show that Wertheimer was right in arguing that the notion of proof does not separate appropriate from inappropriate analyses. But, they show more generally that there is no way at all, using Wertheimer’s internal structure criterion or any other, to select appropriate analyses without considering what particular generalization demands will be met in future.

Choice of Analysis in Structure Mapping. These area examples pose the same problem of selection for structure mapping as for EBG. The existence of multiple analogies involving the same target domain is familiar: Electric current is like fluid flow, electric current is like a stream of particles. Equally,
the same base domain can participate in multiple analogies, though this is less common. The parallelogram above is just an example of this: a jigsaw puzzle piece is like a parallelogram because it has a protuberance that fills a gap, a chevron is like a parallelogram because it is made from a rectangle by sliding rows. The analysis of the parallelogram that is selected will determine what figures can be handled by structure-mapping.

Choice of Analysis in Other Methods. The same point, that more than one analysis of an example is possible, with different generalizability, arises in EBL, PUPS, and synthetic generalization as well, but in different guise. Their application to Wertheimer’s parallelogram problem calls for a different packaging of the problem solution. In EBG and structure mapping, the concept to be generalized can be stated as “figures for which multiplying height by width gives the area” (suppressing some details of how height and width are to be determined for figures other than rectangles.) The example from which the generalization is to be constructed is a parallelogram. The way in which the parallelogram is transformed to a rectangle in order to compute its area, that is, gap-filling or sliding, appears in the analysis of the example, either in the proof that the parallelogram belongs in the concept (for EBG) or in the description of the critical structure of the parallelogram, for structure mapping.

But EBL, PUPS, and synthetic generalization the problem is most naturally recast in such a way that the example is a description of a procedure and its outcome. The generalization mechanism has to deliver a new procedure when given a modified outcome. When packaged in this way, an example in the parallelogram problem is something like “The procedure ‘Transform the parallelogram to a rectangle by gap-filling, then calculate the area of the rectangle’ produces the area of the parallelogram.”

EBL would handle this example by fitting to the procedure schemata for gap-filling, for finding the area of a rectangle, and for finding the area of a novel figure by transforming it into a shape whose area can be found. It would generalize the example to the jigsaw shape, presuming that the gap-filling schema could be instantiated using that shape. It would fail to generalize to the chevron because the chevron could not be used to instantiate the gap-filling schema.

PUPS can attempt to generalize the example procedure to the jigsaw shape, successfully, or to the chevron, unsuccessfully, simply by substituting each shape for parallelogram in both outcome and procedure. Similarly, synthetic generalization can analyze the example as revealing that there is an operation “fill gaps,” and an operation “calculate area of rectangle,” and that applying these operations in sequence will find the area of a figure.

In contrast to EBG and structure-mapping, this presentation of the problem for EBL, PUPS, and synthetic generalization places in the example the way in which the parallelogram was viewed. Thus, the two ways to handle
the parallelogram show up as two examples, one showing gap-filling and one showing sliding, rather than a single example with two different analyses, as in EBG or structure mapping. But Wertheimer's point still emerges: there are different ways to understand how to find the area of a parallelogram, and they support different generalizations.

Is it Possible to Choose a Good Analysis? In Wertheimer's view there are better and poorer analyses of a problem, providing more or less scope for generalization. If this were true, the development of generalization mechanisms could face the problem of selecting good analyses head on. Researchers could pursue Wertheimer's ideas about internal cues that distinguish good from bad analyses, and possibly specialize their generalization machinery to handle only good analyses. But the parallelogram problem shows that the situation is not that simple. Analyses are not good or bad, but rather appropriate or inappropriate for a given generalization objective. Choosing a suitable analysis, without knowledge of specific future generalization demands, is impossible.

DeJong and Mooney (1986) note that EBL could be employed to observe and interpret expert behavior. This approach, which could also be applied using other generalization methods, pushes the problem of selecting a good analysis out of the learning system and into the hands of an expert, who may be better able to address it.

The techniques we have been considering may capture some of the "how" of productive thinking: How to understand something so that it can be generalized, how to use that understanding in generalizing. They also capture the "why": Understanding why procedures work supports generalization. But productive thinking apparently has a "what" as well, not captured by these models or by Wertheimer's conception: Exactly what you think about a problem will determine the generalizations you can make. There may be no substitute for advice from a teacher in determining what the useful things to think about a problem are, since the answer depends on what future generalizations you will need.

SUMMARY

A number of generalization methods, drawn from different areas in cognitive science, can usefully be grouped under the heading analysis-based methods. These methods show different ways in which understanding a procedure can be used in generalizing it. In a procedural domain the analysis that these methods require can be provided by causal attribution heuristics. The EXPL model is a feasibility demonstration for these ideas, showing that the results of a causal analysis can be used to drive either of two differing analysis-based generalization methods.
Empirical data suggest that at least some of the causal attribution heuristics included in the EXPL model are used by people in understanding simple procedures. People are well able to generalize even procedures containing meaningless material on the basis of single examples, consistent with analysis-based generalization but not with inductive approaches. Variation in the treatment of unexplained aspects of examples reveals that people are not consistent in generalization method.

Explanation-based methods, which rely on an explicit domain theory, must be extended to model people’s ability to handle meaningless material. This can be done, and the resulting extended models can mimic any of the other methods in a way that can account for people’s inconsistent generalizations within a unified framework. The data do not indicate whether people are using such a unified framework or are simply using more than one distinct method.

What generalizations these methods produce depend on what analysis is chosen for an example. Contrary to Wertheimer’s ideas, it appears that generalizability is not a simple dimension on which some analyses are better than others. Rather, different analyses can support different generalizations with neither subsuming the other. Choosing among such analyses requires foreknowledge of what generalizations will be needed.

Original Submission Date: July 17, 1987.

REFERENCES


