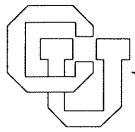
Why and How to Learn Why: Analysis-Based Generalization of Procedures

Clayton Lewis*

CU-CS-347-86 October 1986



University of Colorado at Boulder
DEPARTMENT OF COMPUTER SCIENCE

* This research was sponsored by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research, under Contract No. N00014-85-K-0452, Contract Identification Number, NR 702-009. Approved for public release; distribution unlimited. Reproduction in whole or part is permitted for any purpose of the United States.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; Distribution unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CS-CU-347-86	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION University of Colorado 6b. OFFICE SYMBOL (*If applicable*)	7a. NAME OF MONITORING ORGANIZATION Personnel and Training Research Programs Office of Naval Research (Code 1142PT) 800 North Quincy St.
6c. ADDRESS (*City, State, and ZIP Code*) Campus Box 430 Boulder, CO 80309	7b. ADDRESS (*City, State, and ZIP Code*) Arlington, VA 22217-5000
8a. NAME OF FUNDING / SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N 00014-85-K-0452
8c. ADDRESS (*City, State, and ZIP Code*)	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
6 1153N	RR 04206	RR 04206-06	NR 702-009

11. TITLE (*Include Security Classification*)
Why And How To Learn Why: Analysis-based Generalization of Procedures

12. PERSONAL AUTHOR(S)
Clayton Lewis

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
Technical	FROM _____ TO _____	86-8-26	76

16. SUPPLEMENTARY NOTATION
Submitted to Cognitive Science

17. COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUB-GROUP	Learning, Procedural, Explanation, Analogy
05	10		
	08		

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

Computer learners often develop explanations of events they observe during training. Recent work on generalization suggests that explanations may be valuable in permitting learners to develop generalizations from one or a few examples. We explore this idea by describing four generalization paradigms in which explanations play a part: explanation-based generalization (EBG), structure mapping analogical generalization (SMAG), modificational analogical generalization (MAG) and synthetic generalization (SG). We describe a model, the EXPL system, capable of applying MAG or SG to the generalization of simple procedures in human-computer interaction. We present evidence that EXPL's analysis procedure, which constructs explanations as needed by MAG or SG, embodies heuristic principles used by human learners, and that MAG provides a good account of some human generalization, when retention of examples is not a problem.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Dr. Michael Shafto	22b. TELEPHONE (*Include Area Code*) 202-696-4596
	22c. OFFICE SYMBOL ONR 1142 PT

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted.
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Abstract

Computer learners often develop explanations of events they observe during training. Recent work on generalization suggests that explanations may be valuable in permitting learners to develop generalizations from one or a few examples. We explore this idea by describing four generalization paradigms in which explanations play a part: explanation-based generalization (EBG), structure mapping analogical generalization (SMAG), modificational analogical generalization (MAG) and synthetic generalization (SG). We describe a model, the EXPL system, capable of applying MAG or SG to the generalization of simple procedures in human-computer interaction. We present evidence that EXPL's analysis procedure, which constructs explanations as needed by MAG or SG, embodies heuristic principles used by human learners, and that MAG provides a good account of some human generalization, when retention of examples is not a problem.

Introduction

In a series of thinking-aloud studies of word-processor learning (Lewis and Mack, 1983; Mack, Lewis and Carroll, 1983) it was noticed that learners often spontaneously offered explanations of why things happened the way they did. Learners were under no explicit demand to produce such explanations, yet they showed considerable fluency and ingenuity in developing them. Why were they doing this? Lewis (1986b) speculated that the explanations assisted generalization: determining how their actions were related to observed outcomes could be crucial in permitting learners to build new procedures for accomplishing novel tasks.

This speculation meshes well with recent work on mechanisms of generalization under the headings "explanation based learning" (DeJong, 1981, 1983a, b; Kedar-Cabelli, 1985, Mitchell, Keller, and Kedar-Cabelli, 1986, DeJong and Mooney, 1986) and "analogical generalization" (Pirolli, 1985; Anderson and Ross 1986). In these approaches, in contrast with earlier "similarity-based" methods which look for regularities among large numbers of examples (for review see Dietterich and 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.

In this paper we discuss the application of these analysis-based generalization methods to the task of generalizing simple procedures in human-computer interaction. That is, given an example procedure and its outcome, we will use analysis-based methods to obtain new procedures to produce new but related outcomes. We will then consider data that test the extent to which these models

reflect analysis and generalization as practiced by human learners.

Analysis-based generalization

In similarity-based 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 non-examples.) The generalization produced is the conjecture that any item that satsifies this description is an example 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.

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 a 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.

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. 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. While DeJong and Mooney discuss some advantages of the schema approach, and some other improvements to EBG, the differences between these two explanation-based methods are not important to our discussion here, and we will use EBG as a representative of this class of approach.

EBG requires 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 the domain being considered here, procedures for operating computers, learners frequently encounter examples that they cannot explain on the basis of prior knowledge.

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 inadequate to prove that the example accomplishes the observed outcome, and so no generalization is possible in EBG. 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.

We 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. In the meantime we will table EBG as a model of generalization of procedures, and consider other candidates.

Analogical generalization. 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. For example, our analysis of P and O might attribute the appearance of a particular file in O to the presence of a step in P that mentions the name of this file. If a different file appears in O' we can satisfy this relationship by including in X a step mentioning the name of the new file. Let us call this method SMAG, for Structure Mapping Analogical Generalization.

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 into O' we expect that the same transformation should change P into X. Thus we will construct X by modifying P, suggesting the name MAG, for Modificational Analogical Generalization, for this approach. Anderson and Ross' PUPS system (Anderson and Ross, 1986) is an implemented MAG system; similar ideas are discussed in Pirolli (1985) and Dershowitz (1985). We will follow PUPS in our discussion.

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, giving 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".

SMAG and MAG have in common the exploitation of the idea of analogy, and the dependence on an analysis of how a to-be-generalized example works. SMAG embodies this analysis in the structure that is attributed to P and O, and that is then imposed on X and O'. MAG 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 in our later discussion. SMAG 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 MAG will be left unchanged by the modification process, and will survive in X.

Analogical generalization resembles explanation-based generalization in that it can operate on 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 explanation-based generalization because of the requirement for a formal domain theory in which membership in a concept can be rigorously proved; analogical generalization relaxes this strong requirement and pays a price for it.

Synthetic generalization (SG). In earlier work on the role of explanations in learning (Lewis 1986a) the author developed a generalization technique that resembles SMAG and MAG 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 MAG, or by mimicing the structure of an example, as in SMAG. 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 then 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.

SG 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. From a second example it gleans that TYPE "WRITE" specifies a print operation (say) and that TYPE "BROCCOLI" specifies the file BROCOLLI. The examples themselves are discarded; only the information anout primitive pieces

is retained. Given the demand to remove BROCCOLI, it synthesizes the procedure TYPE "DELETE", TYPE "BROCCOLI" by putting together TYPE "DELETE" and TYPE "BROCCOLI".

The principles underlying SG 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, and Lowry 1983). Winston et al. use auxiliary examples, called precedents, to establish connections between physical features and functional properties; these connections correspond to SG's connections 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 SG's synthesis process, but rather constructs efficient recognition rules for constellations of features that might be observed in other examples.

Mechanism vs. superstition. A key point about SG is that it might produce the procedure TYPE "BROCOLLI", TYPE "DELETE" rather than TYPE "DELETE", TYPE "BROCCOLI". Its knowledge about DELETE and filenames does not include anything about the order in which steps involving them must occur, and the SG procedure does not have access to the original examples from which its knowledge was derived. By contrast, MAG will rarely reorder an example, because a new procedure is always obtained by substituting parts in the example. Only in the special circumstance that a substitution interchanges parts would reordering occur.

A similar contrast emerges in the treatment of unexplained parts of a procedure. In SG, an unexplained step will never be included in a new procedure, because

the synthesizer will have no description of its effects. In MAG an unexplained part of a procedure, that is, one that has no role, will in general be left unchanged in the modification process.

Let us call SG a *mechanistic* process, in that generalizations include only features of examples that are understood, and MAG a *superstitious* process, in that features of examples that are not understood are carried forward into generalizations. Under this definition SMAG is a mechanistic 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. Mechanistic generalization will not perform well when a complete analysis of how an example works is not available.

Analysis of examples

All of these methods require information about the roles of parts of an example. Where does this come from? In the procedure-learning context, how does a learner glean from observing an example like TYPE "DELETE", TYPE "EGGPLANT" what the parts contribute to the outcome? The thinking-aloud studies mentioned earlier (Lewis and Mack, 1982; Mack, Lewis, and 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.

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 and 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 your 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 and Baillargeon 1982), by which all events are assumed to have causes.

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 MAG or SG from an example. Thus combining the EXPL analyis with MAG or with SG provides 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. There appears to be no reason why the EXPL analysis could not drive SMAG, but this has not been done. We will discuss in the following sections those aspects of EXPL pertinent to the examples considered in this paper; complications and extensions needed to handle 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.

Insert Figure 1 about here

This primitive encoding scheme has many limitations; it cannot represent relationships among entities within an event, such as the information that a collection of entities all appear on the same menu, for example. But it has proved adequate to support the analysis of examples of moderate 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.

Previous action. When any components of a system response 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 succession 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 obligatory previous action heuritic 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 we will discuss 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. 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 processing of the example in Figure 1. Note that EXPL's attributions agree well with an intuitive interpretation of the English version in Figure 1.

Insert Figure 2 about here.

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.

Generalization

Using MAG to generalize an analyzed example. To support MAG the results of EXPL's analysis must be converted to the form assumed by the MAG 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 MAG 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.

Insert Figures 3a, 3b, 3c, and 3d about here.

If a part has no substitution, but does have a role specified, MAG 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 SG to generalize an analyzed example. SG requires the results of EXPL's analysis to be cast in a different form. The links shown 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, SG selects from its data base of links actions which will contribute the needed components. Figure 4b shows the resulting procedure.

Insert Figures 4a and 4b about here.

Adding substitution to SG. The example just discussed shows how SG can combine the analysis of two examples to build a new procedure. If only one example is available EXPL's version of SG uses a simple substitution scheme to generalize the single example. Components are assigned to classes, as part 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 SG can be seen as the inclusion of part of the MAG machinery in the SG framework. Without it, SG is unable to generalize many procedures without using links derived from other examples.

Empirical tests of premises of these models.

How well do these EXPL-MAG and EXPL-SG models, or a hypothetical EXPL-SMAG model, account for the behavior of people in analyzing and generalizing examples? While the EXPL analysis heuristics are based in a general way on observations of human learners more specific tests of the use of these heuristics by people are needed. Similarly, evidence is needed regarding whether MAG, SMAG or SG can account for generalizations constructed by people.

To gather such evidence 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.

Mechanistic vs. superstitious generalization. As discussed above, superstitious generalization will normally preserve order of steps, while mechanistic 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 mechanistic generalizer will show the opposite handling: only interpreted steps can appear in a generalization, since steps will be included 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.

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. Estimates 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. 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.

Insert Tables 1 and 2 about 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 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 correllate 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 exaplained. 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, given below 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 and Rohlf 1981) was used to test for differences in response frequencies.

Results and discussion.

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 inter-group differences, except in the case of item RABBIT. Accordingly, results are pooled across groups except in that case.

Insert Table 3 about here.

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 df, $p<.001$).

Insert Table 4 about here.

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 outcome, but there are other possible explanations of the outcome. It is possible that participants assume that items type 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-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 Form1 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.

Insert Figures 6a, 6b, 6c, 6d, and 6e and Table 5 about here.

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 mechanistic 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 consistent with superstitious generalization, it could

also occur if participants have learned that order of steps is generally important in computer procedures and apply that knowledge to the item. 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.

Insert Table 6 about here.

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

Insert Table 7 about here.

When participants assigned roles to 'c43' they treated it appropriately in the generalized procedure, consistent with all of the generalization models 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 mechanistic or superstitious generalization were consistent by participant, participants should fall mainly in the "will work, drop" cell, for mechanistic, or the "order bad, keep" cell, for superstitious generalization. To the contrary, more participants fall in the other two cells, indicating inconsistency across the two items. The "will work, drop" cell is empty, indicating that no participants were consistently mechanistic, while some were consistently superstitious and others were superstitious on one example and not the other.

Insert Table 8 about here.

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 MAG but cannot be handled in SG 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 those 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 the preponderance of 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.

Insert Table 9 about here.

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 mechanism? While the pattern of results is mixed, and does not indicate consistency across items within participants, it appears that responses consistent with superstitious generalization are more common than those indicating mechanistic generalization. It is possible that this finding is dependent on 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 mechanistic methods, which could work with even fragmentary recall of examples, might be more prevalent.

EBG revisited. The ability of participants to generalize examples that contain arbitrary, never-seen-before tokens, as in LADDER or FISH, bears out our earlier contention that EBG, at least as characterized by Mitchell et al. (1986), cannot provide a complete account of learning in this domain. Participants cannot possess domain theories adequate to construct proofs about nonsense elements like "c43".

To attack this problem the EBG framework might be extended to include addition 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 a theory about the significance of the steps in the example, rather than as links or role assignments as needed by SG or MAG. The generalization process itself would work just as it does in normal EBG, but of course the results would no longer be rigorously justifiable, being only be as good as the heuristically-conjectured domain theory.

How would such an extended EBG model compare with SMAG, MAG or SG? Would it be mechanistic or superstitious? The behavior depends on the nature of the domain theory. With appropriate domain theories EBG 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 EBG 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 SMAG 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 EBG framework, SG appears as a special case of SMAG. While SMAG can incorporate arbitrary relationships among attributes of procedures and their outcomes, SG's synthesis process requires that only general principles of combination, and specific descriptions of parts, are permitted. Consequently the domain theory for SG 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 SG. The a priori domain subtheory is an explicit statement of the assumption underlying EXPL's SG planner, without the substitution scheme. The part-specific subtheory contains relationships posited by the analyzer in processing examples. As required for pure SG, 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 mechanistic approach the step c43 is dropped. As expected, 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.

Insert Figures 7a and 7b about here.

It might appear that a superstitious generalization mechanism like MAG could not be accomodated in the EBG framework. After all, one of the functions served by proofs in EBG is filtering features with roles from features without roles, while MAG simply retains features without roles. Nevertheless, with an appropriate domain theory EBG can mimic MAG, at least in simple cases.

The domain theory needed for MAG is somewhat different from those for SMAG or SG. While theories for these models will describe the role of all relevant parts of procedures, the theory for MAG may not. Instead, the MAG 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 b show the treatment of Item FISH in a MAG-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 SG, and are detected 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.

Insert Figure 8a and 8b about here.

Roles of explanation in generalization. Consideration of these generalization mechanisms reveals that explanations can play more than one part in generalization. In SMAG, SG and EBG, explaining the outcome of a procedure leads to the construction of hypotheses about the role of the procedure's parts. These hypotheses are packaged as input to a planner, in SG; as input to a mapping routine, in SMAG; or as extensions to the domain theory in EBG. As long as the hypotheses about the parts are correct, and the theory about the behavior of combinations of parts is correct, new procedures built from the parts will work as

expected. That is, a complete and correct explanation of how the example works enables us to build new procedures that will also work.

In MAG what is explained is not how the procedure determines the outcome, but how the outcome determines the procedure. That is, explanations in MAG are used as a guide to changing the example procedure, given that the desired outcome has changed. The validity of the modified procedure rests not just on the validity of the analysis of the example but also on the validity of the modification rules, which are not part of the explanation of the operation of the original example. It can also rest on the action of unexplained aspects of the example which are left unchanged by the modification rules.

In EBG, SMAG, and SG an explanation serves to filter relevant procedure attributes from irrelevant ones. In MAG relevant features may be explained or not; explanations play no filtering role but are only used to guide modification.

In EBG only, explanations are further used to streamline the construction of generalized procedures by simplifying the application of the domain theory to cases similar to the example. This does not seem to be crucial in the simple procedural examples we have considered here, though they are in other domains considered by Mitchell et al.

Dependence on background knowledge. 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 SG) 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 that we 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 we 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, though 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 and mistakes. There is nothing in that that is limited to the human-computer interaction domain. But it is possible that learners will apply loose-ends without making even 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.

Summary. We can now collect the above arguments, and the indications in the data, and draw conclusions about the generalization processes participants used. Pure SG cannot account for participants' ability to generalize from a single example, though SG plus substitution can. Neither of the mechanistic methods, SG or SMAG, can account for the tendency of participants to reject variant order in the STAR item, or the fact that some participants retain uninterpreted features of examples in generalizing. MAG, a superstitious mechanism, appears to be able to account for the findings in a natural way, except that some participants were willing to accept variant orders in STAR.

All of these generalization mechanisms could be brought within the EBG

framework described by Mitchell et al. But a mechanism like EXPL's analyzer would be needed to build domain theory extensions from examples if EBG is to operate in this domain, since participants clearly were able to generalize examples for which they had no adequate a priori theory.

The results bearing on the analysis of examples support the idea that devices like EXPL's identity and loose-ends heuristics do play an important role, and that participants are able to use them to explain the role of parts of unfamiliar procedures. Such principles of analysis seem to be capable of supporting a variety of different generalization mechanisms.

Returning to our original question, why explain things during learning? It appears that explanations organize knowledge about procedures in a way that supports generalization. Our data suggest that participants used heuristic rules to associate parts of procedures with aspects of outcomes in such a way as to determine what part of a procedure to change to obtain a modified outcome.

REFERENCES

Anderson, J. R. and Thompson, R. (1986). Use of analogy in a production system architecture. Paper presented at the Illinois Workshop on Similarity and Analogy, Champaign-Urbana, June, 1986.

Bullock, M. , Gelman, R., and Baillargeon, R. (1982). The development of causal reasoning. In W.J. Friedman (Ed.), *The Developmental Psychology of Time*. New York: Academic Press.

DeJong, G. (1981). Generalizations based on explanations. *Proceedings IJCAI-7*, Vancouver, 67-69.

DeJong, G. (1983a). Acquiring schemata through understanding and generalizing plans. *Proceedings IJCAI-8*, Karlsruhe, 462-464.

DeJong, G. (1983b). An approach to learning from observation. Proceedings of the 1983 International Machine Learning Workshop, Urbana IL.

DeJong, G. and Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning* **1**.

Dershowitz, N. (1986). Programming by analogy. In R.S.Michalski, J.G. Carbonell & T.M. Mitchell (eds.), *Machine Learning: An Artificial Intelligence Approach, Volume II*. Los Altos, CA: Morgan Kaufmann.

Duncker, K. (1945). On problem solving. *Psychological Monographs*, **58**,

Whole No. 270.

Frazer, J. (1964). *The new golden bough* . (T.H. Gaster, Ed.), New York: New American Library.

Gentner, D. (1983). Structure mapping: A theoretical framework for analogy. *Cognitive Science*, 7, 155-170.

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

Lewis, C.H. (1986a). A model of mental model construction. In *Proceedings of CHI'86 Conference on Human Factors in Computer Systems*. New York: ACM, 306-313.

Lewis, C.H. (1986b). Understanding what's happening in system interactions. In D.A.Norman and S.W.Draper (Eds.) *User Centered System Design: New Perspectives on Human-Computer Interaction.* Hillsdale, NJ: Erlbaum.

Lewis, C.H. & Mack, R.L. (1982). Learning to use a text processing system: Evidence from "thinking aloud" protocols. In *Proceedings of the Conference on Human Factors in Computer Systems*. New York:ACM, 387-392.

Mack, R.L., Lewis, C.H., & Carroll, J.M. (1983). Learning to use word

processors: Problems and prospects. *ACM Transactions on Office Information Systems*, **1**, 254-271.

Mitchell, T.M., Keller, R.M. and Kedar-Kabelli, S.T. (1986) Explanation-based generalization: A unifying view. *Machine Learning* , **1**.

Pirolli, P.L. (1985). Problem solving by analogy and skill acquisition in the domain of programming. PhD Dissertation, Department of Psychology, Carnegie-Mellon University, Pittsburgh, August, 1985.

Shultz, T.R. and Ravinsky, F.B. (1977). Similarity as a principle of causal inference. *Child Development*, **48**, 1552-1558.

Sokal, R.R. and Rohlf, F.J. (1981). *Biometry*. San Francisco: Freeman.

Winston, P.H. (1980). Learning and reasoning by analogy. *CACM*, **23**, 689-703.

Winston, P.H. (1982). Learning new principles from precedents and exercises. *Artificial Intelligence*, **19**, 321-350.

Winston, P.H., Binford, T.O., Katz, B., and Lowry, M. (1983). Learning physical descriptions from functional definitions, examples, and precedents. *Proceedings of AAAI-83*, Washington DC, 433-439.

*I thank Mitchell Blake, Steven 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. 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.

Table 1: Test Items.

Item	In picture	Example	Questions
TRUCK Form 1	truck and boat on screen, keyboard	1. Type "67m" on keyboard. 2. Type "truck" on keyboard. ›››››› Truck turns red.	What does Step 1 do?
Form 2	ditto	1. Type "67m" on keyboard. 2. Type "red" on keyboard. ›››››› Truck turns red.	ditto
LADDER	tree and ladder on screen, keyboard	1. Type "NNA" on keyboard. 2. Type "ladder" on keyboard. ›››››› Ladder rotates 45° 3. Type "NNA" on keyboard. 4. Type "da9" on keyboard. ›››››› Tree rotates 45° 5. Type "n6b" on keyboard. 6. Type "da9" on keyboard. ›››››› Tree shrinks to half size.	What would you do to make the ladder shrink?
MANAGERS Form 1	blank screen, keyboard	1. Type "display3". 2. Type "n25". ››››› System shows list of managers' salaries.	Which step would you change if you wanted a list of managers' ages instead of managers' salaries? Which step would you change if you wanted a list of clerks' salaries instead of managers' salaries?
Form 2	ditto	1. Type "n25". 2. Type "display3". ››››› System shows list of managers' salaries.	ditto
STAR	words alpha, beta, gamma, epsilon in bar at top, star in lower part of screen	1. Touch the star. 2. Touch "beta". 3. Touch a place near the left side of the screen. ›››››› The star moves to the left side of the screen.	If I tried to move the star to the bottom of the screen this way: Touch "beta". Touch the star. Touch a place near the bottom of the screen. Would it work? If not, why not?
FISH	hat and fish on screen, keyboard	1. Type "delete" on the keyboard. 2. Type "c43". 3. Type "hat". ›››››› The hat disappears.	What does Step 2 do? What would you do to make the fish disappear?
RABBIT	rabbit and carrot on screen, keyboard	1. Type "rabbit". 2. Type "remove". 3. Type "HJ4". ››››› Rabbit disappears.	What does Step 3 do?

Table 2: Order of items in test booklets for groups

Group A (n=13)	Group B (n=15)	Group S (n=31)	Group T (n=31)
STAR	STAR	STAR	STAR
TRUCK (Form 1)	TRUCK (Form 2)	TRUCK (Form 1)	TRUCK (Form 2)
TRAIN	TRAIN	TRAIN	TRAIN
LADDER	LADDER	LADDER	LADDER
FISH	FISH	RABBIT	FISH
PERSON	PERSON	MANAGER (Form 1)	MANAGER (Form 2)
HOUSE	HOUSE	PERSON	PERSON
		HOUSE	HOUSE
		FISH	FISH

Table 3: Participant's responses.

Item	Number of responses		Category of response
TRUCK	Form 1	Form 2	
	0	22	step specifies truck or object
	30	3	step specifies red or color
	1	9	step specifies location
	13	12	other
	0	0	no agreement
LADDER	70		'n6b ladder'
	20		other
	0		no agreement
MANAGERS			
First Question	Form 1	Form 2	
	6	6	step 1
	25	22	step 2
	0	3	other
	0	0	no agreement
Second Question			
	25	23	step 1
	5	6	step 2
	0	2	other
	0	0	no agreement
STAR	19		says will work
	53		says will not work because order is wrong
	8		says will not work because order is wrong and gives a reason why order is important
	6		says will not work but does not fit above
	1		none of above
	3		no agreement
FISH			
First Question	9		step 2 does nothing
	26		don't know or can't tell
	53		step 2 is given some role
	1		other
	1		no agreement
Second Question	8		'delete fish'
	57		'delete c43 fish'
	25		other
	0		no agreement
RABBIT	Group S	Group T	
	0	3	nothing
	1	2	don't know or can't tell
	30	25	does something
	0	1	other
	0	0	no agreement

Table 4. Interpretation of step in Item TRUCK.

Content of Step 2	Interpretation of Step 1	
	color	object or location
truck	30	1
red	3	31

Table 5. Responses to two forms of Item MANAGERS.

Answers to questions	Form 1	Form 2
step1, step 1	2	2
step1, step 2	3	4
step 2, step1	23	20
step 2, step 2	2	2

Table 6: Relationship of acceptance of variant order in Item STAR with experience.

Response to new order in STAR	Reported computer experience: less than 55 hours	Reported computer experience: more than 55 hours
will work	8	8
order bad, no reason given	23	23

Table 7. Intepretation and treatment of extra step in Item FISH.

Interpretation of 'c43'	Treatment of 'c43' in new procedure: keep	drop
given role	32	1
no role or role not known	23	7

Table 8: Comparison of responses to Items FISH and STAR.

Response to new order in STAR	Treatment of 'c43' in FISH: no role, keep	no role, drop
will work	7	0
order bad, no reason given	12	7

Table 9. Comparison of role assignment in Items FISH and RABBIT.

Interpretation of 'c43' in FISH	Interpretation of 'HJ4' in RABBIT: given role		no role or role not known	
	Group S	Group T	Group S	Group T
given role	21	16	1	1
no role or role not known	9	9	0	5

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.

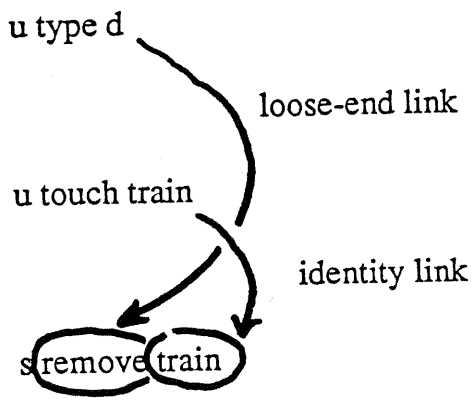


Figure 2: EXPL analysis of example in Figure 1.

Outcome of [[*type d*], [*touch train*]] is [*remove train*].
Role of [*type d*] is [*specify remove*].
Role of [*touch train*] is [*specify train*].

Figure 3a: EXPL output as provided to MAG for example in Figure 1.

Old outcome is [*remove train*].

New, desired outcome is [*shrink train*].

Substituting *shrink* for *remove* maps old outcome to new outcome.

Figure 3b: Determining mapping in MAG.

---

u type r
u touch car
s shrink car

Results of EXPL analysis of auxiliary example:

Outcome of [[*type r*], [*touch car*]] is [*shrink car*].
Role of [*type r*] is [*specify shrink*].
Role of [*touch car*] is [*specify car*].

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

Original procedure:

[[*type d*],[*touch train*]]

Substitution does not apply to [*type d*].

But role of [*type d*] is [*specify remove*].

Substitution transforms this to [*specify shrink*].

Analysis of auxiliary example shows that [*type r*] plays this role.

[*type r*] replaces [*type d*].

Substitution does not apply to [*touch train*] or its role.

Resulting modified procedure is [[*type r*], [*touch train*]].

Figure 3d: Applying substitution of *shrink* for *remove* to the example.

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 auxiliary example in Figure 3c.

Outcome: [*shrink train*]

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

Figure 4b: Procedure constructed for new outcome by using links in Figure 4a.

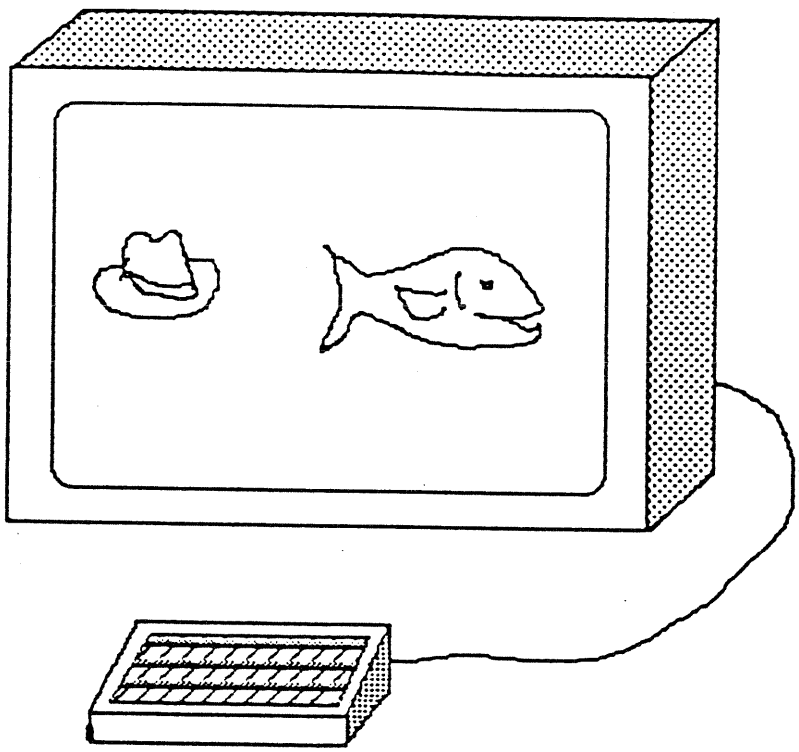


Figure 5: Picture for Item FISH.

Form 1	Form 2
u type display	u type n25
u type 3	u type display
u type n25	u type 3
s show managers' salaries	s show managers' salaries

Figure 6a: Encoding of Forms 1 and 2 of MANAGERS item.

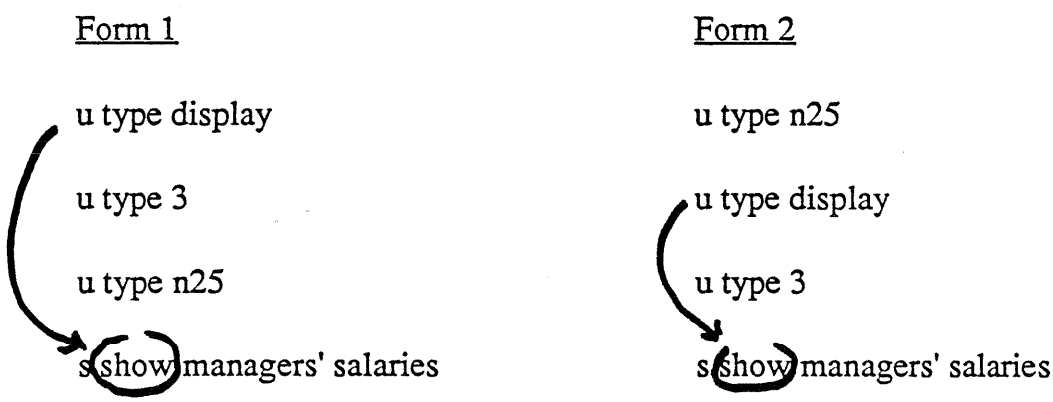


Figure 6b: After placement of identity links.

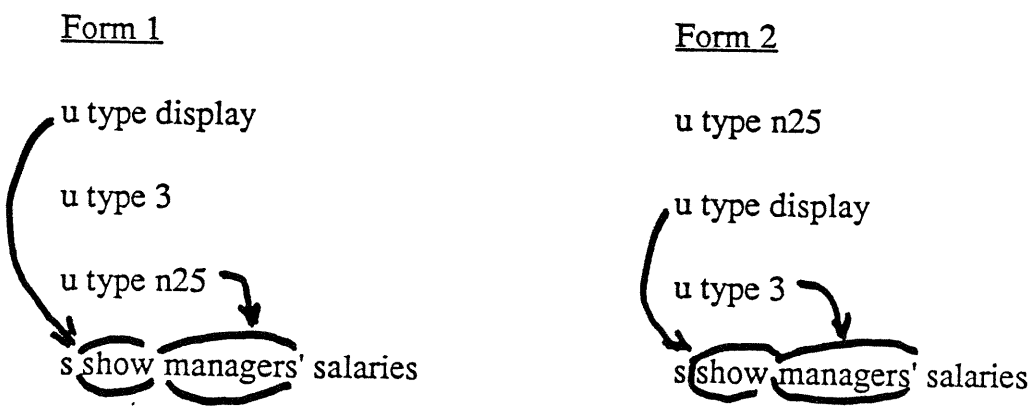


Figure 6c: After applying obligatory previous action heuristic.

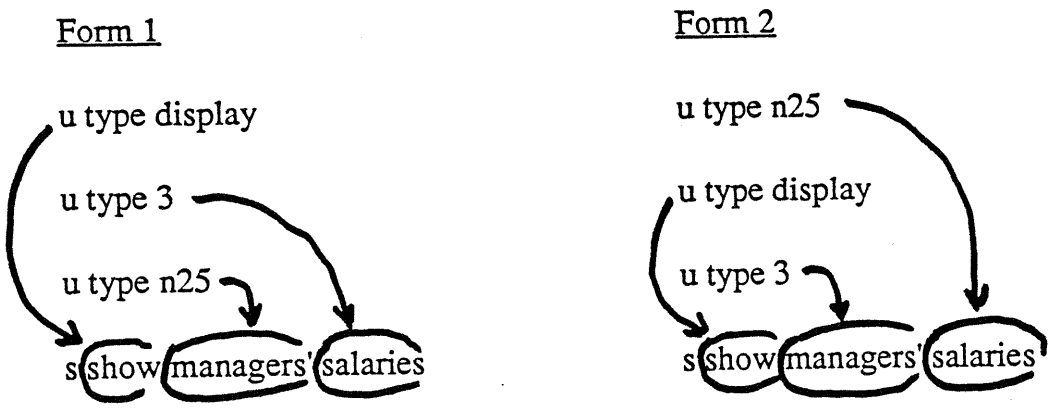


Figure 6d: After applying loose-ends heuristic.

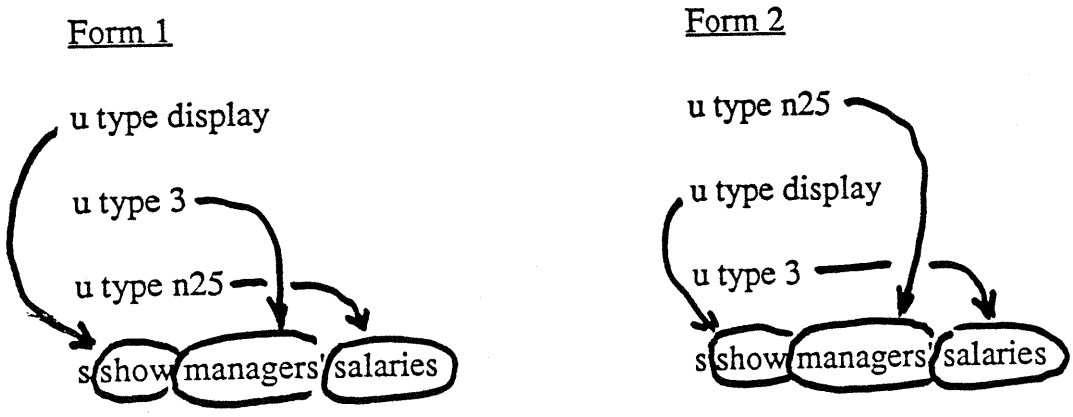


Figure 6e: Result of eliminating obligatory previous action heuristic.

A priori domain theory:

A is an aspect of the result 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 Example1:

[*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 SG-like generalization for Item FISH.

Goal Concept 1:

Procedures P such that *remove* is an aspect of the result 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 result of Example 1.

Generalization based on proof:

P is in Goal Concept 1 if [*type delete*] is a step of P.

Goal Concept 2:

Procedures P such that *fish* is an aspect of the result 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 result of Example 2.

Generalization based on proof:

P is in Goal Concept 2 if [*type fish*] is a step of P.

Construction of procedure to accomplish [*remove fish*]:

Desired procedure P lies in intersection of Goal Concepts 1 and 2.
If [*type delete*] is a step of P, and [*type fish*] is a step of P, P will be in Goal Concepts 1 and 2. Note that [*type c43*] is not included in the construction.

Figure 7b: Continuation of Figure 7a.

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: MAG-like generalization in EBG.

Proof that the example and its outcome satisfy the goal concept:

Let X = [*type delete*]

Y = [*type hat*]

Q = *remove*

R = *hat*

W = *hat* .

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

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

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

Generalization based on proof:

Replacing *hat* by a variable, and leaving other terms in the example fixed, we find that any procedure

[[*type delete*], [*type c43*], [*type* Z]]
and outcome
[*remove* Z]

are in the goal concept.

Therefore to get [*remove fish*] use [[*type delete*], [*type c43*], [*type fish*]].

Figure 8b: Continuation of Figure 8a.