Access and Use of Previous Solutions In A Problem Solving Situation

Jeremiah M. Faries
Brian J. Reiser

Cognitive Science Laboratory
Princeton University

Abstract: An important component of problem solving is the ability to make use of previous examples. This requires noticing the relevance between the current and previous problems. We examine the role of the superficial and structural relations among problems and the remindings that these similarities elicit in a problem solving situation. Students learned to program in an electronic book environment in which they were able to store and later retrieve solved problems. Their use of previous solutions suggests that novices are indeed sensitive to structural similarities and can use retrieved solutions in new problem situations.

Most models of problem solving assume that well learned cognitive skills rely on procedures that have been generalized through use to apply to a variety of problems that may differ on irrelevant features (e.g., Anderson, 1983; Laird, Rosenbloom, & Newell, 1986). Before these procedures have been sufficiently generalized and tuned, however, problem solvers in the early stages of learning a domain may rely on retrieving a previously solved problem and then modifying the solution to fit a current problem (LeFevre, 1987; Ross, 1984; Pirolli & Anderson, 1985). In contrast to procedural models of problem solving, expertise has also been characterized as the ability to access and modify previous solutions to fit new problems (e.g., Kolodner, 1987; Hammond, 1986; Ashley & Rissland, 1987). In these case-based reasoning models, even experts in a domain may rely on modifying a previous solution rather than applying general procedures.

Novices in a domain present an interesting challenge for models of case-based reasoning. Learning about a domain provides the knowledge to organize and encode the features of a problem important to the structure of its solution, so that it can be retrieved later in appropriate circumstances. Experts, however, may need to rely less on remindings than novices, who have not yet formed generalized procedures. The ability to retrieve relevant previous solutions is particularly crucial for novices in a domain. Yet novices may be misled by unimportant superficial similarities between the problem under current consideration and previously encountered problems.

Several recent studies have examined the conditions under which case-based remindings are likely to occur. Some studies have shown that the surface features of a given problem or story have a major influence on the possibility of being reminded (e.g., Ratterman & Gentner, 1987; Ross, 1987). Ross found that the superficial characteristics of word problems may play a major role in the memory processes that lead to these kinds of remindings, at least for novices. Similarly, Ratterman and Gentner found that superficial similarities between stories rather than structural similarities accounted for the large majority of cases in which subjects reported that one story reminded them of another story. One interpretation of these findings is that novices lack the knowledge to appropriately identify and/or categorize problems in a way that would foster appropriate structurally based remindings.

Indeed, structural and surface features are often correlated in the real world (Ratterman & Gentner, 1987) so accessing previous cases by relying on superficial characteristics is arguably a reasonable strategy for generating possible analogues, particularly if the novice lacked the ability to identify the structural components of a problem. It is not clear, however, that most novices are restricted to a strategy that reduces to "a shot in the dark". While it may be true that novices lack concepts and knowledge required to accurately categorize types of problems, it may not be true that they, therefore, rely on superficial characteristics to encode solutions in memory. Even novices may have enough knowledge to partially identify important features of the problem based on the problem solving goals, and should, in principle, be able to use these features to probe their own memories for related problems. This is not to say that surface characteristics of a problem will be unimportant in a reminding episode or that they won't be involved in retrieval in some way. Rather, the claim is that remindings in a problem solving situation will be more often guided by structural characteristics than by surface characteristics.

Hammond (1986) has criticized Gentner's argument on the basis that subjects in these experiments were not given specific problem solving goals. Understanding a story may not provide a well defined goal setting to encourage the use of structurally based remindings. There was no requirement to use the information in these stories, e.g., to make predictions about a current story or solve some type of problem, so it is difficult to specify the clearly structural aspects of the stories or the clearly superficial aspects. Indeed, Seifert, McKoon, Abelson and Ratcliff (1986) have demonstrated the importance of task goals in eliciting remindings.

Ratterman & Gentner's findings may be partly explained by the uncertainty the subjects may have had regarding the importance of the features they were using as a basis of the reported remindings. In a problem solving task where the nature of the task and goals are well-defined, the so-called superficial aspects of the problem are peripheral to the solution, hence it seems likely that remindings will infrequently be guided by superficial similarity alone. The motivation for the present research is to provide a problem solving situation within which structural remindings will be clearly useful, and in which the relevant aspects of the problems are clear. In this way it will be possible to assess the abilities of novices to notice similarities of varying degrees of structural correspondence and to assess the importance of superficial characteristics for access. The well defined nature of the task allows us to clearly determine which aspects of the problem descriptions may be superficial and which may be structural.

The present study is designed to examine remindings of previous solutions in a problem solving situation to test out the propensity and abilities of subjects to notice and make use of correspondences between problems. Subjects in this experiment learn to solve computer programming problems in an electronic book environment. Each problem they solve is stored away on the computer. Students can specify when they want to see any of their previous solutions to help them solve a current problem. By providing the opportunity for the subject to benefit from noticing correspondences, and by using an electronic book environment to trace problem solving and problem retrieval activities, we can investigate the kinds of similarities between problems that subjects noticed and decided to act upon.

THE BATBOOK ENVIRONMENT

We designed an electronic book environment, the Behavioral Analogy Tracing Environment (BATBook), to monitor subjects' use of previous solutions. In this environment, subjects read a text book on the computer screen, compose a solution in an editor, test the solution in an interpreter, and then file away the solution. The subject can request to see a previous solution at any time. In addition, it is possible to search the record of the interactions with the interpreter, and to search through the book for a particular target. The BATBook environment makes explicit a large proportion of the student's problem solving behavior, thereby providing a rich record for analyzing when and how subjects access the written instruction and their previous work.

In the present experiment, novices used BATBook to read the first two chapters of *Essential LISP*, a text book on the LISP programming language (Anderson, Corbett, & Reiser, 1987). Each chapter of the text contains several short sections of instruction followed by problems that apply the knowledge introduced in the section. In addition to the regular problems in the text, we added a second set of problems to Chapter Two. These problems were designed so that each problem contained a cover story similar to a previous problem and was structurally similar to a different problem from the first half. We were interested in whether subjects would retrieve solutions from the first half of the chapter while working on the second half, and whether these retrievals would be governed by surface or structural similarities. In addition, we included two variations on the method for storing previous solutions. One group provided a verbal label for each solution they constructed and later could use this verbal label to retrieve a previous solution, while the other group did not label their solutions and could retrieve them only by referring to the problem description or content of the solution.

The BATBook environment runs on Sun workstations. The screen contains a Text Window, an Exercises Window, a LISP Interpreter Window, and a Problem Submission window that appears when the subject stores a completed solution (see Figure 1).

Reading the Text: The left half of the screen contains the Text Window, which displays approximately one page of text from the book. When the student finishes reading the current page, he or she selects the "page forward" button (labeled "+") using the mouse. Subjects can page backward (the "-" key) and can return to the first page of the chapter (the "Page 1" key). The design of the Text Window was loosely based on the Superbook electronic book environment (Remde, Egan, & Landauer, 1987). When subjects reach the last page of the section, they are instructed to begin working on the problems associated with that section, and are not permitted to page forward until they have completed that problem set.

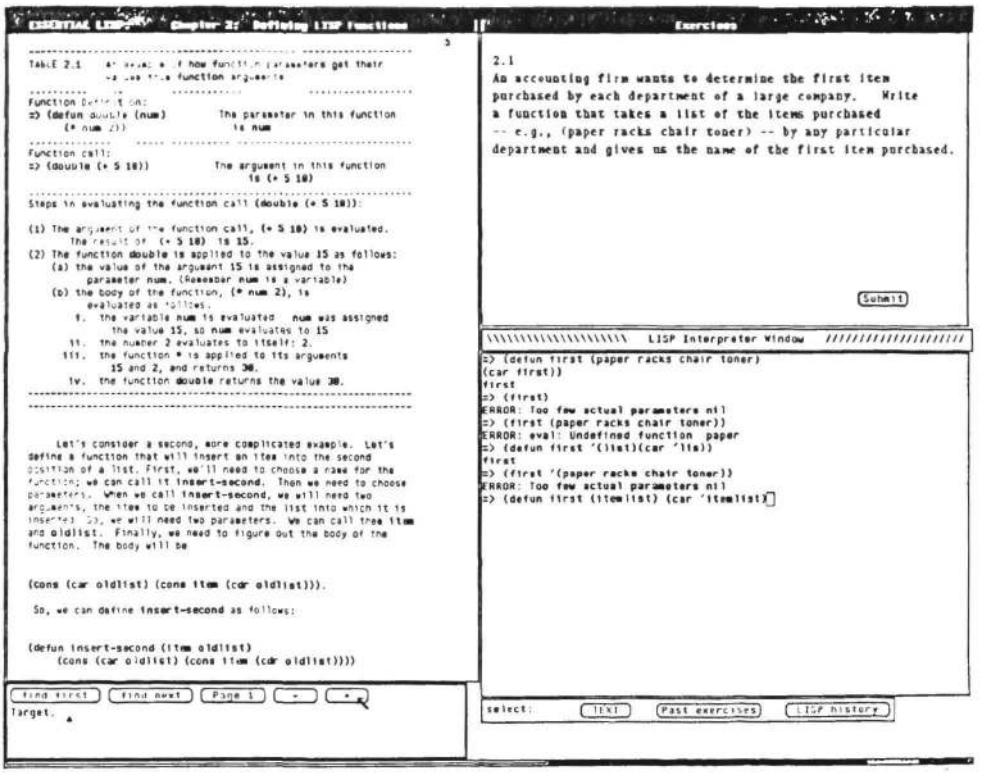

Fig 1. The BATbook screen layout showing text, problem description, and LISP work.

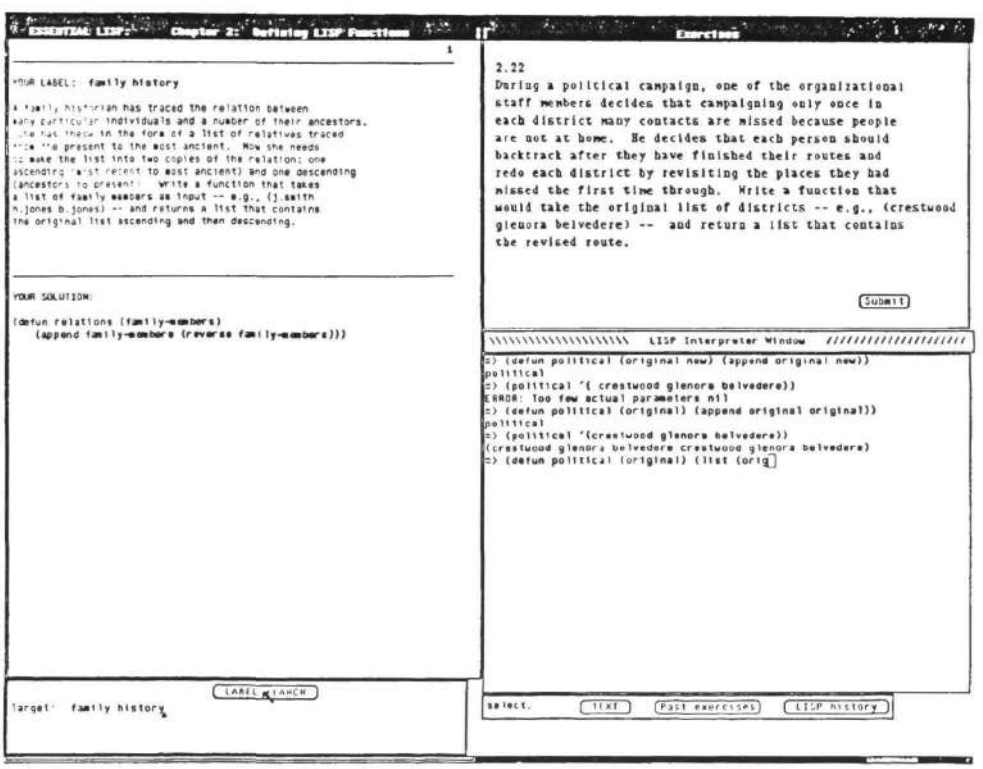

Fig 2. The retrieval of a previous solution using a label generated by the subject.

Searching the Text: Subjects can search through the current chapter of the text using a search capability similar to that provided in most word processing programs. To do this, the subject types a target of one or more words and then selects the "Find First" or "Find Next" key, which searches the text for the first occurrence or next occurrence of the search string. Thus, if the subject wishes to find a particular portion of text, he or she can page through the text or use the Text Search feature to find it.

Constructing a Solution: When the subject completed the reading for a section, he or she was instructed by the program to begin the associated problems. The problem sequence was initiated by clicking a "Start Problems" button in the Exercises Window, upon which the first problem for the section was displayed in the window. Problems required writing LISP function calls (Chapter One), and defining functions (Chapter Two). Subjects typed their function calls and function definitions into the LISP Interpreter Window which contained a Common LISP interpreter. Subjects could change a function definition by using a simplified Emacs editor invoked from the LISP Window. The editor contained commands to move left and right one character, up and down one line, delete a character, delete a line, and insert a deleted line. The editor also contained a commands to save the function and enter the revised definition into LISP, and to abort the edit with the definition unaltered. Typically subjects typed a definition into the LISP Window, tried their function on some examples, then edited the definition until it worked properly.

Submitting a Completed Problem: When the subject considered the solution to be correct, he or she submitted the problem in order to store the solution. This was initiated by clicking the "Submit" button in the Exercises Window. At that point, BATBook prompted the student to enter the function name and then checked the student's solution. If the solution was incorrect, the student was informed of the example for which their function computed an incorrect result and asked to try to fix their solution. Subjects were required to attempt to fix their function and submit a second solution. If the solution was still incorrect, the subject was again informed but this time was given the option to continue with the next problem, or to continue trying to fix the solution.

Labeling the Solution: One group of subjects (Label Group) was asked to label their final submitted solution (whether correct or incorrect) for the problem. The subject was asked to type in a brief label for the solution. Subjects were told that the label could be used at a later time to retrieve the problem description and the solution. The other group of subjects (Non-Label Group) were not asked to label the problem, and could later refer to the problem using a probe from either the problem description or the content of the solution itself. Following the completed problem, a new problem was displayed in the window. Upon completion of the problem set the subject was instructed to resume reading the text.

Searching Previous Solutions Using Labels: When a subject in the Label group wanted to look at a previous solution to help with a current problem, he or she clicked the "Past Exercises" key in the LISP Interpreter Window. At that point, subjects typed in part or all of a previous label and clicked the "Label Search" key to initiate the search. If a problem with a matching label was found, the complete problem description and the subject's final solution to the problem were displayed in the Text Window (see Figure 2). After finding a previous solution, the subject typically returned to the LISP Interpreter Window to attempt to map something from the solution to the current problem, or decided to search for a different problem if the search retrieved something that the subject then decided would not be useful. Subjects were informed that they might find it useful to search for previous solutions. There was no limit on the amount search that the subjects could do.

Searching Previous Solutions Using Keywords: When a subject in the Non-Label Group wanted to search for a previous solution, he or she clicked the "Past Exercises" button, and was prompted to type in one or more words to use as a search target. The subject then clicked the "Find First" or "Find Next" as in search of the Text Window. The first or next occurrence of the search string was sought among the problem descriptions and their solutions. If a match was found, the problem description and the subject's solution was displayed in the Text Window, just as for a Label Search. Only the problem and description for a single problem was displayed at a time.

Searching the LISP Interaction History: In addition to searching the text or previous solutions, subjects could also search the log of their interaction with the LISP Interpreter. We included this feature because we expected that subjects might find it useful to retrieve an episode of testing or debugging a function, in addition to retrieving their final answer to a problem. Search of the LISP history was initiated by selecting the "LISP History" button, whereupon subjects were prompted to type their search string as in the Keyword search. The search found the first or next occurrence of the search string in the log of the student's interaction with the LISP interpreter for that chapter. The log was displayed in the Text Window

with target string in inverse video.

At the conclusion of each problem, the display of the Text Window returned to the page that instructed them to continue with the problem set so that previous solutions, LISP history, or text would no longer be displayed. The subject would be required to search again to retrieve desired information.

Procedure: Subjects were run individually in three two hour sessions. The last two sessions were always on consecutive days. Subjects were free to explore any available aspect of the environment during the session. The LISP Interpreter was available throughout the entire session. The system required the subject to read each section and submit the associated problems before going on to next section. Subjects worked on Chapter One in the first session, and on Chapter Two in the second session. The third session concerned the "review" problems, which contained surface and structural similarities to the earlier problems in Chapter Two. No additional reading was presented in the third session.

Materials: The problems of Chapter One were included unchanged from the *Essential LISP* text. Cover stories were added to the 11 original questions of Chapter Two, the "source" problems. Each question was constructed using a different cover story. These cover stories were plausible scenarios for the problems, but were unrelated to the structure of the solution. The third session contained the 11 "target" problems. Each target problem was constructed so that it contained the same cover story as one of the source problems, and was structurally similar to a different source problem. Thus, it would be possible for each target problem to retrieve a previous problem focusing either on surface or on structural similarity. An example target problem and its surface and structural source problems is shown here:

Target: During a political campaign, one of the organizational staff members decides that when campaigning only once in each district many contacts are missed because people are not at home. He decides that each person should backtrack after they have finished their routes and redo each district by revisiting the places they had missed the first time through. Write a function that would take the original list of districts -- e.g., (crestwood glenora belvedere) -- and return a list that contains the revised route.

Structural similarity source: A family historian has traced the relation between many particular individuals and a number of their ancestors. She has these in the form of a list of relatives traced from the present to the most ancient. Now she needs to make the list into two copies of the relation: one ascending (most recent to most ancient) and one descending (ancestors to present). Write a function that takes a list of family members as input -- e.g., (j.smith h.jones b.jones) -- and returns a list that contains the original list ascending and then descending.

Surface similarity source: A political campaign organizer is making up lists of neighborhoods for the campaign workers to visit. He would like to have his workers visit the districts at different times of the day so that people who are not home at particular times of the day may be reached on another day. To do this he plans to have his campaign workers first visit the district they visited last on the day before, and then continue in the same order as they did previously. Write a function that takes a list of district names -- e.g., (crestwood glenora belvedere) -- and returns a list with the last district name moved to the beginning.

Subjects: Subjects were 10 Princeton University Students, community members, and research staff, who were paid $5 per hour for their participation. Subjects were selected who had no formal training in a computer language and very little or no informal programming experience. The subjects were semi-randomly placed in each condition with consideration given to extreme math SAT scores in order to balance these individuals between conditions. The average SAT score was 670 in the Label condition and 690 in the Non-Label condition.

RESULTS AND DISCUSSION

Throughout all three sessions there were many cases in which subjects referred to previous problems, past work in the LISP History, and previous portions of the text. We will focus our analyses on the search for previous work during the review half of Chapter Two (day 3). It was in this set of problems that every problem had one structural and one superficial counterpart in the initial set. There were an average of 6.4 cases of searching behavior in Chapter One and 5.3 cases in the first half of Chapter Two.

Type of Problem Retrieved: Subjects in the review half of Chapter Two exhibited an average of 6.5 cases of search behavior. We categorized the searches of solutions and LISP History according to whether the retrieved problem was the superficial antecedent, structural antecedent, or a problem unrelated in the design. Of the total searches, 49% retrieved a structurally similar problems while only 9% retrieved a problem with superficial similarities. The large difference supports our claim that novices are indeed sensitive to structural correspondences, and can exploit this recognition to use their solutions in later problems. The remaining 42% of the retrieved problems were of unspecificed similarity.

These include cases in which the subject rejected the retrieved problem and searched again with a different cue, and cases in which their solution or the errors they encountered were similar in ways not captured in the design.

Considering the two groups separately, the Label subjects exhibited an average of 10.4 reminds. Of these searches, 57.7% retrieved structural counterparts, 5.8% retrieved superficially related problems, and 36.5% were unspecified. Subjects in the Non-Label condition exhibited far fewer instances of search behavior, only 2.6 cases per subject. Of these only 15% represented searches for structurally related problems, 23% represented searches for superficially related problems, and 62% represented searches for unspecified problems.

It is important to consider the relations between those retrieved problems that were neither structural nor surface antecedents. There may have been structural correspondences between problems or the difficulties encountered in the solution other than those between problems designed to be structurally similar. Therefore, we considered the events that precipitated the searches for previous solutions. Interestingly, 78% of the structural remindings occurred immediately after reading the problem description. 19% were precipitated when the subject encountered an error, and 3% occurred after trying LISP calls prior to attempting a function definition. For surface related searches, 50% followed problem descriptions, 33% followed errors and 17% occurred after the problem was solved. Of the unclassified searches, 41% followed problem descriptions and thus provided no clear indication of the reason for the search. An additional 37% of the searches were due to extensive trial and error searching by one subject who had used numbers as his solution labels. Finally, 22% of the unclassified searches followed errors, and almost always retrieved a previous portion of the LISP History in which a similar error occurred. Thus, these searches that retrieved "unrelated" problems in fact recovered solutions to problems that differed on superficial features but bore an important structural similarity in the types of difficulties encountered in constructing the solution. Interestingly, none of the subjects whose search was preceded by an error actually used part of the error message to find a corresponding part of the history. Instead, these typically used a function or variable name related to superficial characteristics of the problem.

Type of Search Key Used: Subjects were free to construct their own probe words for the search. We categorized the search probes used by the subjects into those that referred to structural and surface features of the problem. Surprisingly, only 23% of the probes were related to the structural features of the problem, whereas 77% were related to superficial characteristics. The environment may have biased the use of superficial features, since the most distinctive strings in the targets were usually related to the superficial aspects of the problem. Nevertheless, it is important to stress that subjects could often locate a structural correspondent using a superficial aspect of the event. For example, one subject, after reading the target problem description shown in Figure 2, searched his LISP history for the word "family" which was the name of the function he had defined on the previous day for the structurally isomorphic problem.

The surface information is evidently accessible and can be used to identify structural correspondences. The amount of structurally based search suggests that people organize memory for events in ways that reflect structural and functional importance. The use of superficial search keys suggests that superficial information is retained, although it appears that access to this information was through structural routes. It is even possible that superficial information helps to to cohere the structural aspects of the memory. Although it might be argued that subjects remembered events using surface features as cues and them made assessments of structural soundness to decide whether to use the event, the short latencies between problem presentation and initiation of structural searches make this alternative unlikely.

We also considered the proportion of searches in which subjects successfully utilized the retrieved information. Success was defined as a correct solution following a search; this occurred on 62% of the searches. Failures included cases in which retrieving a previous solution led to an incorrect solution to the current problem, and those cases in which the retrieved information was rejected in order to initiate another search. These comprised 38% of the searches. A structural probe led to a successful search 67% of the time, whereas it led to a failure 33% of the time. A superficial probe led to a successful search 57% of the time, whereas it led to a failure 43% of the time. The failures also include behavior such as using trivial labels (one, two, three) to "page" through the problem histories. This occurred in two episodes and accounted for 39% of the so-called failures. It should also be noted that many of the failures formed part of episodes that led to a successful search in a few steps.

There results suggest that the nature of the probe used did not affect the probability of successfully retrieving useful information. Surface features

can be used to retrieve structurally relevant episodes, even though they are not structurally diagnostic. They are a part of the memory for the event that is salient, but not necessarily influential in the selection of isomorphs. We also considered the type of labels, function names, and variable names generated by subjects. Here again, many of these labels and function names referred to surface features of the problem, yet this did not prevent these labels and solution components from being effective search cues to retrieve structurally related problems.

Conclusions: We have presented evidence that people in a clearly defined problem solving situation are sensitive to and are able to make use of structural correspondences between problems. The subjects were rarely misled by the superficial correspondences and were able to identify and make use of structural similarities between problems. They may not have had all the well formulated rules they needed, but they were sensitive to the structural nature of the problems and could, therefore, detect functional similarities. The efficient use of examples requires subjects to have encoded the relevance of particular examples, and to remember enough about them to generate successful retrieval descriptions.

As Ratterman and Gentner (1987) have demonstrated, superficial characteristics may be important for access, but this may not necessarily reflect how novices store information for previous problems. The availability of superficial information does not imply that the structural information is unavailable. In fact, it appears that the structural organization of the memories for the problems solved may have been responsible for the activation of the surface level features. The use of superficial probes to locate structurally similar problems demonstrates that surface features are not dissociated from information used to make inferences between related problems. Surface features may simply be more salient and easier to specify than more abstract features. This salience does not necessarily interfere with problem oriented memory organization and retrieval.

ACKNOWLEDGEMENTS

We are grateful to Eric Ho and Antonio Romero for assistance in programming BATBook, and to Dennis Egan, Louis Gomez, Tom Landauer and Joel Remde of Bellcore for discussions of their Superbook program. The research reported here was supported in part by contract MDA903-87-K-0652 from the Army Research Institute, and by a research grant from the James S. McDonnell Foundation to Princeton University. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Army or the McDonnell Foundation.

REFERENCES

Anderson (1983). *The architecture of cognition.* Cambridge, Mass.: Harvard Univ. Press.

Anderson, J. R., Corbett, A. T., & Reiser, B. J. (1987). *Essential LISP.* Addison-Wesley.

Ashley, K. D., & Rissland, E. L. (1987). Compare and contrast, a text of expertise. *Proceedings of AAAI-87, Sixth National Conference on Artificial Intelligence*, Seattle, WA.

Hammond, K. (1986). The use of remindings in planning. *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Amherst, MA.

Kolodner, J. L. (1987). Extending problem solver capabilities through case-based inference. *Proceedings of the Fourth International Conference on Machine Learning*, Irvine, CA, Morgan Kaufman, 167-178.

Laird, J., Rosenbloom, P., & Newell, A. (1986). *Universal subgoaling and chunking* Boston, Mass.: Kluwer.

LeFevre, J. (1987). Processing instructional texts and examples. *Canadian Journal of Psychology*, 41, 351-364.

Pirolli, P. L., & Anderson, J. R. (1985). The role of learning from examples in the acquisition of recursive programming skills. *Canadian Journal of Psychology*, 39, 240-272.

Ratterman, M. J. & Gentner, D. (1987). Analogy and similarity: Determinants of accessibility and inferential soundness. *Proceedings of the Ninth Annual Conference of the Cognitive Science Society.* 23-35.

Remde, J. R., Gomez, L. M., & Landauer, T. K. (1987). Superbook: An automatic tool for information exploration - hypertext?. *Proceedings of Hypertext '87.*

Ross, B. H. (1984). Remindings and their effect in learning a cognitive skill. *Cognitive Psychology,* **16,** 371-416.

Ross, B. H. (1987). This is like that: The use of earlier problems and the separation of similarity effects. *JEP: LMC*, **13**, 371-416.

Seifert, C. M., McKoon, G., Abelson, R. P., & Ratcliff, R. (1986). Memory connections between thematically similar episodes. *JEP: LMC*, **12**, 220-231.