COGNITION AND INSTRUCTION, 1990, 7(4), 323–342

Goal Setting and Procedure Selection in Acquiring Computer Skills: A Comparison of Tutorials, Problem Solving, and Learner Exploration

Davida Charney
Department of English
The Pennsylvania State University

Lynne Reder and Gail W. Kusbit
Carnegie Mellon University

This study investigated the relative benefits for acquiring computer skills of (a) learner-initiated versus experimenter-supplied goals and (b) active practice at selecting and applying procedures versus directed execution of procedures. Subjects with varied computer experience but no knowledge of electronic spreadsheets were randomly assigned to an interactive instruction or an exploration-learning condition. Both groups read identical descriptions of 12 spreadsheet commands. The exploration-learning group experimented with commands at will, setting goals, and selecting and applying procedures. The interactive instruction group did not set their own goals but, instead, worked three training problems per command. The training problems for six commands were presented without solutions; subjects solved them by actively selecting and applying procedures and afterwards received feedback. For the remaining six commands, the training problems were tutorials with explicit solutions that subjects typed in verbatim. Two days after training, all subjects solved 12 test problems. Learning commands by solving problems without explicit solutions led to longer training times but also significantly faster and more successful performance at test than either tutorial training or exploration learning. Exploration learning did not differ significantly from tutorials in training time or performance at test. Regression analyses indicated that the advantage of problem solving was not due simply to longer training times.

With the increasing need or desire among learners of all ages to acquire computer skills, greater attention is being devoted to the form that such in-

Requests for reprints should be sent to Davida Charney, Department of English, Pennsylvania State University, 117 Burrowes Building, University Park, PA 16802.

struction should take in manuals and guides for computer users. Various positions are represented in the literature on instruction in computer skills, including some advocating long, highly elaborated, user manuals, some favoring on-line interactive tutorials, some calling for drastic reductions in manual content, and others hoping to do away with manuals altogether (Carroll, 1984; Scharer, 1983; Shrager & Klahr, 1983; Tausworthe, 1979). Because using a computer is a special case of skill learning in general, the designers of computer manuals might be inclined to consult the literature on skill-learning strategies. Unfortunately, however, this literature does not provide clear-cut guidance on this issue, because empirical evidence can be marshaled in support of several of these seemingly contradictory options, which appear under such labels as *learning by example, problem solving,* and *discovery learning.*

We conceive of initial skill learning as consisting of three critical components: (a) learning the functionality of a novel procedure, (b) learning the conditions under which the procedure is appropriately applied, and (c) learning how to execute the procedure (Charney & Reder, 1987). In other words, learning a skill means knowing what procedures exist for accomplishing various goals, recognizing the circumstances in which to select a particular procedure, and knowing how to apply or execute the procedure within a given situation. However, these three components of skill learning are generally not all supported in the learning strategies most commonly manifested in computer manuals.

LEARNING STRATEGIES

Learning by Example

Instructional texts that promote learning by example typically contain numerous worked-out examples that learners study and use as models for solving problems on their own. A worked-out example consists of a problem or goal statement and an explicit sequence of steps to a correct solution. In addition, the text may provide formal rules that generalize the solution as well as negative examples illustrating situations in which the rule is not applicable or has been applied incorrectly. Worked-out examples help learners categorize problems with similar solutions and construct solutions to novel problems by analogy to the example (Anderson, Farrell, & Sauers, 1984; Nitsch, 1977; Sweller & Cooper, 1985).

The learning-by-example approach can be found in many computer user manuals, specifically those that illustrate features or commands with examples of how they are applied in specific situations. When such examples are well chosen, they can be highly beneficial; computer users have been shown to perform on-line tasks better after studying manuals with examples rather than unelaborated manuals (Reder, Charney, & Morgan, 1986). However, although examples provide information relevant to the three components of skill learning, they do not provide opportunities for learners to actively practice the second and third components, selecting and applying procedures. The relative passivity of learning by example poses a motivational problem; computer users are known to dislike reading manuals and to prefer instead to do things (Carroll, 1984; Scharer, 1983). More important, in current theories of skill learning, active practice has been shown to be essential for transforming declarative knowledge into procedural knowledge (Anderson, 1983). Learning by example may help learners form appropriate declarative representations of concepts, procedures, and problem situations in the domain. However, it may not provide sufficient independent practice at selecting and applying procedures, opportunities to "practice the integrated act" in a motivating, learning-by-doing environment (Brown, 1983).

Interactive Tutorials

Interactive tutorials, at least as they are commonly represented in commercial user manuals, appear to offer the advantages of learning by example without its passivity. Like examples, tutorials present the learner with a problem and a series of steps to a correct solution. Instead of simply reading and analyzing the worked-out problem, however, the learner enters the steps of the solution at the computer and can observe the computer's prompts and feedback messages. Given the similarities between tutorials and worked-out examples, one might expect tutorial users to learn at least as much as those who simply read manuals with examples and to gain motor and perceptual practice at executing the commands. However, learners following tutorials are apt to enter the keystrokes mechanically, without thinking about the purpose of each action or even observing the screen (Carroll, 1984). Further, although following a tutorial may be more motivating than simply reading a manual, learners still lack opportunities to practice selecting among commands or to practice generating appropriate instantiations of commands for particular situations.

Problem Solving

Problem solving as a learning strategy grows out of the tradition of chapter-end exercises in math and science textbooks. Like the learning-by-example and tutorial approaches, this approach starts with a text containing problems prepared in advance for the learner. Instead of studying or carrying out a solution provided in the text, however, the learner must arrive at a solution independently by retrieving candidate procedures from memory, selecting appropriate ones for the given problem, generating correct instantiations, and executing the procedures correctly. In addition, learners may receive feedback on the solution in the form of answers in the back of the book. Like worked-out examples, problems provide information about the skill domain: By examining and comparing problems, the learner can draw inferences about the typicality of various problem situations and can store correct solutions to be used as models.

The efficacy of problem solving may depend heavily on the choice of problems and how they are presented. When problems involving a particular procedure only appear at the end of the chapter in which that procedure is introduced, learners may be practicing not selection but only application of procedures. Further, as Sweller and Cooper (1985) argued, learners may spend time fruitlessly on incorrect solution paths and may, therefore, fail to acquire good models of solutions. At this point, we are unaware of any commercial computer manual that promotes a problem-solving learning strategy.

Discovery Learning

In its long history, the term *discovery learning* has been applied to a wide range of learning strategies, including a few of those just discussed (for reviews, see Hermann, 1969; Wittrock, 1966). The term has come to mean instruction that is less direct or less explicit than standard instructional texts (i.e., instruction that leaves something for the learner to discover). Discovery learners are commonly encouraged to formulate general rules or principles inductively from examples supplied to them, but both Hermann and Wittrock emphasized that deductive reasoning can be considered discovery learning as well. Generally, researchers have found that discovery learning tends to take more time than expository or rule-based learning and that it promotes transfer of learning to new tasks rather than simple retention of studied materials. In addition, effective discovery learning requires a fair amount of guidance (Scandura, 1964).

Recent advocates of discovery as a method for skill learning (Carroll, Mack, Lewis, Grischkowsky, & Robertson, 1985; Kamouri, Kamouri, & Smith, 1986) have defined it largely in terms of learner-initiated exploration (in fact, they tend to use the terms *exploration-based training* or *guided exploration*). As in more familiar forms of discovery learning, exploration requires learners to induce rules or procedures. Rather than working with examples provided in an instructional text, however, learners in exploration-based training experiment directly on a device or computer application, in effect creating their own examples and setting their own goals. In addition, the learners may control the pace and sequencing of their activities. Carroll et al. argued that cognitive skill learning will be more successful when learners set their own goals and figure out how to accomplish them, because such goals are intrinsically more motivating and are less susceptible to misinterpretation than goals set in an instructional text.

EMPIRICAL COMPARISONS OF LEARNING STRATEGIES

Few studies have directly compared these strategies for skill learning. Sweller and his associates (Sweller, 1988; Sweller & Cooper, 1985; Tarmizi & Sweller, 1988) conducted several studies comparing learning by example and problem solving in the domain of algebra and argued that studying worked-out examples was more beneficial. Problem solving, they contended, interferes with the acquisition of problem-type schemata because the standard means–end problem-solving strategies typically used by novices impose a heavier cognitive load than studying worked-out examples. Consistent with this analysis, Sweller and Cooper found that subjects who studied four example algebra problems and solved four problems during training were faster and more accurate at solving new problems at test than subjects who trained by solving eight problems.

Charney and Reder (1986; Charney, Reder, & Wells, 1988) compared examples, tutorials, and problem solving as strategies for learning a computer application (with a methodology similar to the study reported here). We found that problem solving required more training time than the other strategies but that commands learned through problem solving were more likely to be selected and applied correctly at test. Although hands-on tutorials might be expected to benefit learners more than examples, we found no difference: Commands learned with either tutorials or examples were equally likely to be used correctly at test, and test performance in either condition was significantly poorer than that for the problem-solving condition.

Tarmizi and Sweller (1988), in an extension of Sweller and Cooper's (1985) work, also found limitations to the benefits of studying certain kinds of examples relative to problem solving. In particular, when examples required students to integrate information from several sources (e.g., the text of a geometry problem and its associated diagram), the additional cognitive load required to process the example canceled out its benefit. Their analysis suggests that, because subjects in the Charney and Reder (1986; Charney et al., 1988) paradigm must divide their attention between verbal text and the graphic display on the computer screen, the benefit of the examples and tutorials may have been reduced. Because the example, tutorial, and problem-solving conditions all required subjects to integrate the same amount of information from both the text and the screen, however, it is not clear why this burden would not apply equally across conditions. The inconsistency between the relative benefits of examples and problem solving in our study and those of Sweller and his colleagues may instead arise from differences in the testing procedures. Sweller's subjects were tested immediately after the acquisition phase, whereas we imposed a 2-day delay.

Carroll et al. (1985) compared a commercial tutorial for a word-processing program with a set of guided exploration (i.e., discovery learning) materials. To force learners to discover procedures through interactions with the computer, the guided exploration materials omitted procedural rules or any explicit formal statements of how to execute the commands. The materials did provide hints about useful keys and menus, checkpoints for assessing success, and remedies for recognizing and recovering from mistakes. The learners, who were experienced secretaries, experimented with procedures at their own initiative and formulated their own goals for applying them. Once again, tutorials were ineffective. The guided exploration learners trained more quickly, completed the criterion tasks more quickly, and made fewer procedural errors than the group trained with the tutorial. It is difficult to generalize from the results of this study, however, because the length, content, and presentation format of the instructional information varied widely in the two conditions.

Although Charney and Reder (1986; Charney et al., 1988) found problem solving to be superior to tutorials or examples, and Carroll et al. (1985) found that guided exploration was superior to a tutorial, problem solving and exploration have not been directly compared. Unlike examples and tutorials, both problem solving and exploration-based learning provide learners with opportunities to practice selecting and applying procedures. Problem solving requires the learner to figure out which procedure is most appropriate to solve the problem and then to use it. Effective exploration-based learning would involve these steps, too. The difference between these two instructional strategies, however, is in who sets the goals (i.e., the problems for the learner to solve). In problem solving, the goals are set in the instructional materials. In exploration-based learning, by contrast, the learners generate their own problems and, hence, set their own goals. The relative effectiveness of these strategies may depend heavily on the learners' abilities to set appropriate goals. In Carroll et al.'s study, the learners were secretaries who already understood the situations in which word processing would be used and, hence, were probably well prepared to appreciate the goals of the basic procedures. In that situation, exploration-based learning may indeed be optimal. We suspect, however, that if the goals are less familiar to learners, they may not do as well at generating problems for themselves and may need an instructional manual that provides representative problems for them to solve.

In this study, we explored the relative value of learner-initiated versus instructionally provided goals by assessing the effectiveness of tutorials, problem solving, and exploration for learning to use an electronic spreadsheet. All three instructional conditions aimed to facilitate learners' acquisition of the procedures, their syntactic components, and appropriate conditions for their use. The tutorial was intended as an effective, hands-on, example-based instructional condition. The problem-solving materials provided task-specific goals but required the learner to select appropriate procedures and to execute them independently. The exploration condition was intended to examine the role of learner-initiated goal setting, and it also provided practice at selecting and executing procedures. Subjects in this condition were provided with instructions on how to use the procedures but not with examples or problems on which to practice them. Rather, learners were permitted to invent their own problems. This condition was not intended as a classic discovery-learning paradigm and does not replicate Carroll et al.'s (1985) "guided exploration." In particular, our subjects did not induce the procedures independently through their exploration. Instead, subjects in this condition were presented with the same information about the commands as subjects in the other conditions, except for information about potential goals as implied in the practice problems. This exploration-learning condition allowed us to focus on the relative benefits of learner-initiated versus instructionally provided goals while controlling for effects of the instructional text itself.

Because exploration and problem solving provide active practice at selecting and applying procedures, both should be more effective than tutorial training. Our expectations were less clear-cut for the relative value of exploration and problem solving. If independent goal setting is a critical feature of effective skill learning, then the exploration condition should be more effective than the problem-solving condition. However, if explorers need to be highly familiar with the domain in order to set appropriate goals independently, then we might expect problem solving to be more effective, because our subjects had no prior familiarity with spreadsheets.

METHOD

Design

Subjects learned 12 commands for an electronic spreadsheet (VisiCalc). The study employed a mixed design. The between-subjects variable determined whether subjects learned the commands by experimenting with them at will and inventing their own problems (exploration group) or by working on problems presented in a manual (interactive instruction group). The within-subject variable involved just the interactive instruction group. For this group, the problems for half the commands were presented along with solutions that subjects entered verbatim at the computer (tutorial condition). For the other 6 commands, no solutions to the problems were presented; subjects worked to solve these problems as best they could and then received feedback (problem-solving condition). The tutorial and problem-solving conditions were varied within subjects to allow additional data on problem types to be collected as part of a separate study that is not reported here. Also for this reason, more subjects were assigned to interactive instruction than to exploration. Dependent measures were training time, percentage of responses correct at test, and mean time to a correct solution at test. Exploration learning was compared separately with problem solving and tutorial instruction using analysis of covariance with rated previous computer experience as the covariate.

Materials

The instructional text was a user manual for the VisiCalc electronic spreadsheet consisting of a two-page general introduction and a brief (one-page) description of each of 12 basic commands. These descriptions included formal rules (i.e., keystroke sequences) for generating syntactically correct commands. A sample description of one command ("Move Row or Column") is provided in the Appendix.

In addition to the descriptions, 4 problems pertaining to each command were developed (for a total of 48 problems). Each problem set a specific goal for changing a given spreadsheet and could be solved using 1 of the 12 commands (sometimes applying the same command more than once) in conjunction with standard cursor-control procedures. For any problem, only 1 command provided the best outcome, as measured by efficiency of keystroking, correct final appearance of the spreadsheet, or correct functioning of the spreadsheet. For example, the goal of alphabetizing the entries on a payroll sheet was best solved using the Move Row or Column command rather than retyping all the entries.

Two forms of each problem were developed: (a) a *tutorial* form that presented a step-by-step solution that subjects were to enter verbatim at the computer and (b) a *problem-solving* form that presented a goal that subjects were to achieve on their own. The two forms are illustrated in Figure 1 with a depiction of the spreadsheet as it appeared on the computer. The problem-solving form (Figure 1a) presented a goal that the learner was to try to achieve. Feedback on the optimal solution to the problem appeared on the next page. The tutorial form of the problem (Figure 1b) presented the same goal statement followed immediately by the solution to the problem, that is, a sequence of keys to type in. Both forms of the problem, then, contained the same information about the goal and the optimal solution, differing only in when the solution was presented.

A manual was constructed for each subject with the 12 command descriptions in random order. For each subject in the interactive instruction group, commands were randomly assigned to problem form (tutorial or problem solving), with the constraint that half the commands be assigned to each form. Three of the 4 problems prepared for each command were randomly selected and interspersed in the training manual. The remaining 12 problems were reserved for use as test items. Similarly, for each subject in the exploration group, 1 problem per command was randomly selected as

1a. Training problem in Problem-Solving Form

Alphabetize the names, by putting the rows containing Steele and Stewart further down in the appropriate spots.

Feedback appearing on the following page:

You could have used the following sequence of commands (starting with cell A1 as the current cell) to solve the preceding problem:

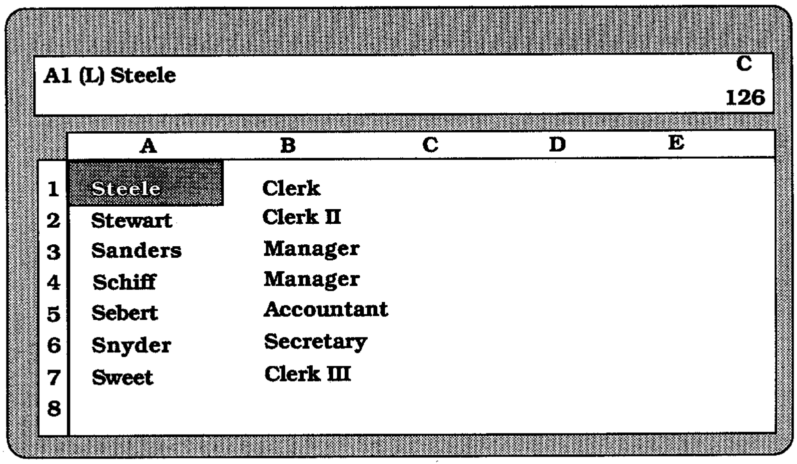
```
/M . A7 [RETURN]
/M . A7 [RETURN]
```

1b. Training problem in Tutorial Form

To alphabetize the names, by putting the rows containing Steele and Stewart further down in the appropriate spots, type the following sequence of commands (starting with cell A1 as the current cell):

```
/M . A7 [RETURN]
/M . A7 [RETURN]
```

VisiCalc display as it appeared on the computer:



```
A1 (L) Steele                                         C
                                                    126
     A           B            C          D         E
1  Steele      Clerk
2  Stewart     Clerk II
3  Sanders     Manager
4  Schiff      Manager
5  Sebert      Accountant
6  Snyder      Secretary
7  Sweet       Clerk III
8
```

FIGURE 1 Sample training problem. From "Designing Interactive Tutorials for Computer Users" by D. Charney and L. Reder, 1986, *Human-Computer Interaction, 2,* p. 307. Copyright 1986 by Lawrence Erlbaum Associates, Inc. Reprinted by permission.

a test item. Test items, presented in random order, were identical to the problem-solving form except that they did not provide feedback.

Procedure

Subjects participated in two sessions, a training session and a testing session 2 days later. In the training session, all subjects worked through the user manual at their own pace.

Interactive instruction group. Subjects in the interactive instruction group worked through each command and its associated problems or tutorials in the order they were presented in the manual. They were instructed not to return to previous pages of the manual. They were presented with 36 training problems, 3 for each command, interspersed through the manual. For half the commands, problems were presented in tutorial form and half in problem-solving form. For problems in tutorial form, subjects were instructed to enter the provided solution verbatim. For problems in problem-solving form, they were told to use whatever procedures they liked to arrive at the goal. After the subjects finished working on a problem (successfully or not), they turned to the next page that provided feedback in the form of the recommended sequence of keystrokes. Subjects were not allowed to turn back to previous pages and, therefore, could not consult descriptions of the commands or previous problems as they worked.

Exploration-learning group. Subjects assigned to the exploration group were provided with the same instructional manual as the other subjects, but the training problems were omitted. Further, subjects were told that there was no constraint on the order in which they could study or practice the commands. These subjects experimented with the commands at their own initiative, looked back in the manual at any time, and freely created their own spreadsheets or modified the set of practice-problem spreadsheets that were stored on-line.

Test session. Two days after training, all subjects were asked to solve 12 problems on the computer (1 for each command studied) without access to the manual and without feedback.

Data collection. Both the training and the test sessions were videotaped to record the subjects' interactions with the computer and to collect reading and solution times. Subjects sat at an IBM Personal Computer (IBM-PC) with the training manual on a lectern beside it. One black-and-white video camera was focused on the manual and another was focused on the computer screen. A mixer connected to a video cassette recorder produced a split image recording the top few lines of the current manual page above the contents of the VisiCalc display. A millisecond timestamp appeared at the bottom of each video frame enabling the collection of training and test times.

Subjects

Sixty-five undergraduates participated for pay and/or course credit. Twenty subjects were randomly assigned to the exploration group and 45 to the interactive instruction group. Previous computer experience was rated with a questionnaire in which subjects specified the types of computers they had used, the programming languages they had studied, and the frequency with which they had used various computer applications (e.g., word processors, graphics packages, statistical packages). No one having experience with electronic spreadsheets was allowed to participate. The response categories on the questionnaire were weighted; subjects' scores ranged from 0 (no experience) to 3. The exploration and interactive instruction groups were essentially equivalent in previous computer experience, each group having a mean score of 1.29 (*SD*s = 0.57 and 0.79, respectively).

RESULTS

Scoring

To measure success at solving a problem, we analyzed the videotapes to see if the subjects had satisfied the goal specified for each problem, using the appropriate command. One point was awarded for each correct solution, 0 points for incorrect attempts, and .5 points for solutions in which the subject attempted to use the appropriate command but missed some minor detail of the command syntax. To measure times for correct or partially correct solutions, we calculated the elapsed time from the frame containing the first appearance of the manual page displaying the problem to the frame in which the last keystroke of the solution was entered. To compare results in the three conditions in comparable ways, we ran separate analyses of variance for each pair of conditions, treating the problem-solving and tutorial learning conditions as a between-subjects rather than as a within-subject variable. We assume, then, that the size of the effects to be reported are conservative estimates, because within-subject analyses would have pulled out additional variability due to subjects.

Training Time

Overall, subjects in the exploration group devoted less time to training than did those in the interactive instruction group (84.1 min vs. 95.5 min, respectively). The extra training time for the interactive instruction group was clearly due to the problem-solving condition, which, not surprisingly, took more time than the tutorials. In terms of the average time that subjects spent per command, as indicated in Table 1, exploration learning was intermediate, taking significantly less time than problem solving, $F(1, 63) = 22.3$, $p < .01$, but not differing significantly from tutorial training ($F < 1.0$).

Ability to Solve Test Problems

As indicated in Table 1, subjects in the interactive instruction group were significantly more likely to solve a test item successfully if they had studied

TABLE 1
Training Time and Test Performance as a Function of Skill-Learning Strategy

	Interactive Instruction		
Phase	*Tutorials*	*Problem Solving*	*Exploration*
Training			
Mean seconds per command	331	562	351
SD	79	189	92
Test			
Mean seconds to correct solution	134	96	108
SD	95	48	44
Mean percentage of correct solutions	.53	.71	.54
SD	.27	.24	.24

the relevant command with problem-solving rather than tutorial training, $F(1, 88) = 11.96$, $p < .01$, replicating the result in Charney and Reder (1986). The success rate for subjects in the problem-solving condition was also significantly higher than for the exploration group, $F(1, 63) = 7.1$, $p < .01$. The success rates for exploration learning and tutorial training did not differ.

Time to Solve Test Problems

The mean times to solve a problem correctly at test are presented in Table 1. Due to missing cells for subjects who failed to solve one or more of the test problems, the solution-time data were analyzed with chi-square tests on solution times above and below the common median. For the interactive instruction group, subjects solved a test problem significantly faster if they had practiced the command with problem-solving rather than tutorial training, $\chi^2(2, N = 87) = 4.07$, $p < .05$. The solution times for the exploration group were intermediate but did not differ significantly from either of the other forms of training.

Computer Experience

Subjects' computer experience scores were used as covariates in the data analyses. Experience only slightly influenced the analysis of problem-solving and exploration data, but the effects were in the expected directions. More computer experience led to marginally less training time per command, $F(1, 62) = 3.4$, $p < .07$; the regression coefficient was negative (-48.4). Further, more experience led to slightly greater success at test, $F(1, 62) = 4.1$, $p < .05$; in this case, the regression coefficient was quite small (0.08). For both measures, however, the adjusted cell means changed very little from those reported in Table 1.

DISCUSSION

Training with problem solving took more time than either learner exploration or tutorial training but produced faster and more successful performance at test. Although exploration learning took more training time than tutorial training, it did not lead to better performance at test. Given that previous studies (Carroll et al., 1985; Kamouri et al., 1986) have found learner exploration to be significantly superior to tutorial training, why did subjects in this study benefit so little from learner exploration?

It is possible that the differences between conditions were due to the amount of effort (i.e., total study time) expended during training rather than to the nature of the training per se. Subjects spent more time on problem-solving training than on tutorial training or exploration, and they were more successful at test for commands they had studied with problem solving than with either of the other conditions. Conceivably, the advantage of problem solving could simply be that it induces subjects to study longer.

To address this possibility, we performed a series of stepwise logistic regressions. First, we considered whether or not study time accounted for performance within a training condition. The dependent measure of primary interest was test accuracy, that is, the subjects' ability to use the command correctly to solve the problem at test. The variables used to predict test accuracy were as follows: command to be learned, subject, computer experience, and time spent during training to practice that command. We did not include the number of training problems solved correctly as a predictor variable, because subjects rarely failed to solve tutorial training problems correctly, and it was difficult to distinguish problem episodes for subjects in the exploration-learning group.

Of the predictor variables, computer experience typically entered into the regression analyses and then exited, because the variability due to experience was subsumed by individual differences among the subjects. Adding the subject variable significantly improved the prediction in all three training conditions: in the problem-solving condition, $\chi^2(44, N = 270) = 96.6$; in the tutorial condition, $\chi^2(44, N = 270) = 111.6$; and in the exploration condition, $\chi^2(19, N = 240) = 69.2$, with $p < .01$ in all cases. Adding the command variable also significantly improved the prediction: for problem solving, $\chi^2(11, N = 270) = 54.0$; for tutorial training, $\chi^2(11, N = 270) = 42.5$; and for exploration learning, $\chi^2(11, N = 240) = 58.3$, with $p < .01$ in all cases.

Of greater interest is the extent to which training time accounted for variability in learning. For the exploration-learning group, training time did not enter the analysis at all. For the problem-solving and tutorial conditions, training time had significant effects, $\chi^2(1, N = 270) = 7.6$ and 7.8, respectively, $p < .01$, but in each case the effects were much smaller than for the previous two predictors. It is interesting to note that the coefficients for these two conditions were also very small and in opposite directions. In the case of problem solving, the coefficient for training time was -0.002; for tutorial training, the coefficient was 0.005. These results appear to reflect the subjects' response to the relative difficulty of the two training conditions. In the case of tutorial training, in which subjects simply typed in what they were told, spending more time on training problems was probably an indication that subjects were trying harder to remember the information. Accordingly, subjects who spent more time were likely to perform better at test. In the case of problem solving, however, spending more time was an indication of the greater difficulty of the command itself. Longer training times usually indicated repeated (and, therefore, unsuccessful) attempts at solving a problem. Accordingly, if a given subject spent more time on a training problem, he or she tended to have difficulty again with that command at test.

These analyses are consistent with an analysis of covariance on subjects' success at solving test problems in the tutorial and problem-solving conditions, in which we used training time as the covariate. When the means for these two groups are adjusted for time spent during study, the advantage of problem-solving training over tutorial training increases. The effect due to training condition remains significant, $F(1, 87) = 25.5$, $p < .01$, such that subjects perform better when trained with problem solving (the adjusted mean percentage correct solutions are .79 for problem-solving and .45 for tutorial training). Training time is also significant, $F(1, 87) = 12.1$, $p < .01$, but the regression coefficient is negative (-0.00010), indicating that subjects who took more time during training solved fewer test problems correctly. We take these results to suggest that differences in study time during training reflected item difficulty.

Finally, and most important, we ran a logistic regression on performance across all three conditions (problem solving, tutorial training, and exploration learning) to compare directly how much variability in test accuracy was accounted for by training condition and how much by training time. In this case, three variables significantly improved the prediction: command to be learned, subject, and training condition, with $\chi^2(11, N = 780) = 119$, $\chi^2(64, N = 780) = 227.4$, and $\chi^2(2, N = 780) = 20.1$, respectively; $p < .01$ in all cases. Training time did not enter into the regression. Taking all these analyses together, it is reasonable to conclude that the superiority of problem solving is not an artifact of the time subjects devoted to the materials during training.

This leaves us again with the question of why we found no evidence of significant benefit from exploration learning. As noted previously, our exploration condition differed from classic discovery-learning paradigms in that our subjects were not asked to induce the syntactic rules for the various procedures. Our study did allow these subjects to set their own goals (problems) and to actively practice selecting and applying procedures. It is conceivable that induction of the procedural syntax rather than goal setting per se is the critical feature of successful discovery learning. It seems more plausible, however, that successful learning depends on a subject's ability to set appropriate goals. As others have noted, in order for discovery learning to work, learners must succeed at discovering the "desired" principles (Anthony, 1973) or at recognizing the potential usefulness of a rule (Scandura, 1964). If exploration learners in our study were unable to invent appropriate situations for applying the commands, they would not have discovered what they needed to learn. The ability to set appropriate goals may be predicated on a subject's knowledge of the domain.

The poor performance of our exploration group, then, may be due to the subjects' minimal prior knowledge of the domain. Carroll et al. (1985), in successfully using exploration learning to teach word-processing skills, employed temporary office workers who knew a great deal about the goals and strategies involved in producing business correspondence. By contrast, our subjects were students who had little experience with the goals and strategies for using a spreadsheet. By working the problems in the manual, the interactive instruction group (who saw training problems in both problem-solving and tutorial forms) surely learned a great deal about how spreadsheets are typically used and in what contexts specific commands may be useful. However, although the exploration group had access to the spreadsheets used for the problems in the manual, they were not able to invent corresponding problems independently, and they did not always choose to use the spreadsheets we had created. In short, the ability of the exploration group to appreciate fully the functions of the various spreadsheet commands, and when to use them, may have been severely constrained.

Informal observations of the activities of the exploration learners during training are consistent with this conclusion. To their credit, these subjects seemed engaged in the task as a whole and were generally successful at generating syntactically correct commands. They tended to practice each command numerous times, whereas subjects in the interactive instruction group were limited to only three opportunities to practice each command (as provided by the three training problems per command in tutorial or problem-solving form). However, the exploration-based learners did not practice (or did not discover) all the suboptions and ramifications of the commands; instead, they repeatedly practiced relatively simple applications. As a result, many failed to notice important consequences of the commands. For example, most exploration subjects were successful at splitting the display into two windows but never attempted to scroll the windows independently. Similarly, most subjects were successful at using the Replicate command to copy data in one set of cells to another location but never used it to reapply mathematical functions to a new subset of data. Finally, exploration learners tended to practice one command all at once and then to move on to a different command, creating in effect a "massed-practice" condition. By contrast, because the training problems for a given command were distributed at various points in the manual, interactive instruction subjects were provided with "spaced practice," which is known to improve skill-learning performance (Charney & Reder, 1986).

Most important, we noted that subjects in the exploration group tended not to invent spreadsheets that represented full-blown meaningful scenarios for manipulating numerical data. Although we provided subjects with a typical spreadsheet at the onset of the experiment as well as a list of other available spreadsheets stored on-line, subjects did not make great use of them. Even when they started with a blank spreadsheet, few subjects attempted to create a checkbook register or a monthly budget ledger. Instead, they tended to create ad hoc fragments, typing in a few columns of numbers chosen apparently at random. As a result, they tended not to create or recognize situations that motivated the use of particular commands. For example, if subjects experimented with the Windows command (which split the display into independently scrollable windows) while using an empty spreadsheet or one containing entries that easily fit on the display, they would not appreciate the value of the command for bringing distant columns or rows into view. In fact, they might not appreciate that the size of the display screen constrains effective use of a spreadsheet. By contrast, the training problems for the Windows command (used in the problem-solving and tutorial training conditions) presented subjects with the goal of bringing distant columns into view in an appropriate context (e.g., a column with year-end totals and a column representing data for the third month).

Exploration-based discovery learning seems to involve two distinct phases, *problem formulation* and *problem solving*. In the problem-formulation phase, learners decide to experiment and invent a task or problem. For learners unfamiliar with the domain, this phase may be difficult or problematic. These learners may have trouble fully exploring the domain, because inexperience prevents them from setting appropriate problem goals. That is, unless they know something about the range of problem situations that may arise in the domain, they may not be able to invent typical or important tasks or problems. Further, novices may not invent situations that allow them to assess the appropriateness of one procedure over another. Both potential problems stem from a lack of knowledge of what is possible or desirable to do in the new domain and a lack of knowledge of typical problem situations that may arise.

In the second phase, problem solving, learners work on the task they set for themselves, using domain-specific techniques for finding solutions. We might expect the problem-solving phase of exploration learning to share some of the same benefits as the problem-solving strategy discussed previously, particularly in providing practice at applying commands appropriately. It might not provide effective practice at selection, however, because the learner may have a particular command in mind when generating a problem. By contrast, learners who are given a problem to solve do not know ahead of time which command is intended and, therefore, must review the available commands and select one that seems appropriate. Further, exploration learners may have more difficulty accurately evaluating their progress, and it is not easy to provide feedback to learners who set their own goals. Discovery learners may retain misconceptions that do not quickly produce salient errors. In addition, learners may not be able to evaluate the quality of their solutions to distinguish between makeshift solutions and more efficient or elegant ones.

CONCLUSION

Our results suggest that, for learning procedural skills, solving problems that are set in an instructional text is more effective than independent goal setting, at least for learners who have little general knowledge of the skill domain. The advantages for problem solving did not accrue strictly from independently practicing the selection and application of procedures. Subjects in both the problem-solving condition and the exploration group practiced selecting and applying procedures independently. What differed was the nature of the problems and the availability of feedback. To invent appropriate problems (i.e., set appropriate goals), learners must know something about typical or important situations in the domain. Learners may already have this knowledge, as Carroll et al.'s (1985) subjects apparently did. Their subjects were experienced secretaries learning to use a text editor. In this case, one might expect them to benefit more from exploration learning than problem solving; additional research on this question is required. When learners do not already have situational knowledge, however, they will probably have difficulty setting relevant goals on their own. This conclusion is consistent with earlier research on elaborations in which computer novices who had advance knowledge of target tasks performed better after studying a short unelaborated manual, whereas those without such information performed better with manuals containing well-chosen examples and other elaborations (Reder et al., 1986). When learners are familiar with task situations that may arise in the domain, they can anticipate how the procedures they study may be usefully applied, and they are less dependent on a text to illustrate situations in the domain.

Problems that are posed in a text provide opportunities for learners to actively select and apply procedures. At the same time, problems can illustrate typical or important contexts of use for various procedures. Although the problems in a text may not be as relevant to individual experiences and goals as those the learner might invent, they do have several other advantages. Problems can be devised to cover the full range of a system's capabilities, to illustrate situations in which some procedures are more appropriate than others, and to anticipate and correct potential misconceptions.

ACKNOWLEDGMENTS

This article was sponsored in part by the Office of Naval Research, Contract Number N00014-84-K-0063, and in part by National Science Foundation Grant BNS-03711 to Lynne Reder.

REFERENCES

Anderson, J. R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.

Anderson, J. R., Farrell, R., & Sauers, R. (1984). Learning to program in LISP. *Cognitive Science, 8,* 87-129.

Anthony, W. (1973). Learning to discover rules by discovery. *Journal of Educational Psychology, 64,* 325-328.

Brown, J. S. (1983). Learning by doing revisited for electronic learning environments. In M. A. White (Ed.), *The future of electronic learning* (pp. 13-32). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Carroll, J. (1984). Designing MINIMALIST training materials. *Datamation, 30*(18), 125-136.

Carroll, J., Mack, R., Lewis, C., Grischkowsky, N., & Robertson, S. (1985). Exploring a word processor. *Human-Computer Interaction, 1,* 283-307.

Charney, D., & Reder, L. (1986). Designing interactive tutorials for computer users. *Human-Computer Interaction, 2,* 297-317.

Charney, D., & Reder, L. (1987). Initial skill learning: An analysis of how elaborations facilitate the three components. In P. Morris (Ed.), *Modelling cognition* (pp. 135-165). Chichester, England: Wiley.

Charney, D., Reder, L., & Wells, G. (1988). Studies of elaboration in instructional texts. In S. Doheny-Farina (Ed.), *Effective documentation: What we have learned from research* (pp. 47-72). Cambridge, MA: MIT Press.

Hermann, G. (1969) Learning by discovery: A critical review of studies. *Journal of Experimental Education, 38,* 58-72.

Kamouri, A., Kamouri, J., & Smith, K. (1986). Training by exploration: Facilitating the transfer of procedural knowledge through analogical reasoning. *International Journal of Man-Machine Studies, 24,* 171-192.

Nitsch, K. E. (1977). *Structuring decontextualized forms of knowledge.* Unpublished doctoral dissertation, Vanderbilt University, Nashville.

Reder, L., Charney, D., & Morgan, K. (1986). The role of elaborations in learning a skill from an instructional text. *Memory & Cognition, 14,* 64-78.

Scandura, J. (1964). An analysis of exposition and discovery modes of problem solving instruction. *Journal of Experimental Education, 33,* 149-159.

Scharer, L. (1983, July). User training: Less is more. *Datamation, 29,* 175-182.

Shrager, J., & Klahr, D. (1983). Learning in an instructionless environment. In A. Janda

(Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems* (pp. 226–229). New York: ACM.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science, 12,* 257–285.

Sweller, J., & Cooper, G. (1985). The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction, 2,* 59–89.

Tarmizi, R., & Sweller, J. (1988). Guidance during mathematical problem solving. *Journal of Educational Psychology, 80,* 424–436.

Tausworthe, R. (1979). *Standardized development of computer software, Part II.* Englewood Cliffs, NJ: Prentice-Hall.

Wittrock, M. (1966). The learning-by-discovery hypothesis. In L. Shulman & E. Keislar (Eds.), *Learning by discovery: A critical appraisal* (pp. 33–75). Chicago: Rand McNally.

APPENDIX A

A Typical Manual Entry for a Difficult Command: MOVE ROW OR COLUMN

The Move command moves the entire row or column that contains the current cell to another position on the work sheet.

Procedures

/M [FROM] . [TO] [RETURN]	Moves the contents of row or column in the [FROM] coordinate to the row or column specified in the [TO] coordinate.

The Move command requires the following information:

1. *The FROM coordinates:* The coordinates of a cell in the row or column that you wish to move. VisiCalc automatically fills in the coordinates of the current cell (e.g., D5) as the FROM coordinates. If the current cell is not in the row or column you wish to move, type [BKSP] to erase these coordinates and type the coordinates of a cell in the row or column you want to move. Then type a period. Three periods appear on the edit line. Now you can type the "TO" coordinates.
2. *The TO coordinates:* The coordinates of a cell specifying the destination of the move. The TO coordinates must contain either the same column letter or the same row number as the FROM coordinates. The VisiCalc program determines whether to move a row or a column by comparing FROM and TO coordinates: If the column letter in the two coordinates is the same, then a row is moved; if the row number is the same, then a column is moved.
3. The difference between the FROM and TO coordinates tells VisiCalc

where to put the moved information. If the FROM coordinates (e.g., D5) have the same column letter as the TO coordinates (e.g., D3), then the contents of Row 5 will move up to Row 3. If the FROM coordinates (e.g., D5) have the same row number as the TO coordinates (e.g., B5), then the contents of Column D will move left to Column B.

VisiCalc makes room for the row or column you move by shifting the other rows and columns over. So moving a column or row to a new location does not "cover up" any other entries.