Complexity Management in a Discovery Task.

Christian D. Schunn
Department of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213
schunn@cmu.edu

David Klahr
Department of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213
klahr@cmu.edu

Abstract

Previous psychological research about scientific discovery has often focused on subjects' heuristics for discovering simple concepts with one relevant dimension or a few relevant dimensions with simple two-way interactions. This paper presents results from an experiment in which subjects had to discover a concept involving complex three-way interactions on a multi-valued output by running experiments in a computerized microworld. Twenty-two CMU undergraduates attempted the task, of which sixteen succeeded, in an average of 85 minutes. The analyses focus on three strategies used to regulate task complexity. First, subjects preferred depth-first to breadth-first search, with successful subjects regulating the number of features varied from experiment to experiment most effectively. Second, subjects systematically regulated the length of their experiments. Third, a new explicit search heuristic (Put Upon Stack Heuristic) used by successful subjects is described.

One of the most complex cognitive tasks that humans face is scientific discovery. It combines the mystery of creativity with the rigor of experimentation and hypothesis testing. By studying the psychological processes involved in scientific discovery, one can test the scalability of psychological theories developed using simpler tasks, and develop theories of the integration process of numerous subprocesses, which does not occur in simple tasks.

The first psychological investigations of scientific discovery used simple rule discovery tasks (e.g., Bruner, Goodnow, & Austin, 1956; Wason, 1960). Subjects had to discover a rule for classifying instances as either members or nonmembers of a concept which was an arbitrary concatenation of a few simple features. Simon & Lea (1974) proposed that rule discovery could be viewed as search in two problem spaces: the rule space (the set of all rules for classification of instances) and the instance space (the set of all instances to be examined). Rule discovery is comprised of the set of processes, algorithms, and heuristics for searching each of the two spaces, and integrating the search between the spaces.

Klahr and Dunbar (1988) extended the dual search idea by proposing that scientific discovery can be understood in terms of search in two problem spaces: the hypothesis space and the experiment space. A primary difference between scientific discovery and rule discovery is that the relationship between the hypothesis and the experiment is very straight-forward in rule discovery tasks, whereas it is very complex in the real scientific discovery process (Klahr & Dunbar, 1988). To study the psychological processes in situations with this more complex relationship, some researchers have used computerized microworlds (e.g., Mynatt, Doherty, & Tweney,1977; Klahr & Dunbar, 1988; Dunbar, 1989, 1992).

As the complexity of the domain grows, it becomes increasingly necessary to use heuristics to simplify the search in the two spaces. Klahr & Dunbar (1988) describe two heuristics associated with two subgroups of subjects in their experiments: their "experimenters" limited their search mainly to the experiment space and their "theorists" limited their search mainly to the hypothesis space. Klahr, Dunbar, & Fay (1990) further identified several search heuristics used by subjects for searching within each of the two spaces. Example heuristics identified for searching the experiment space included designing experiments which maintain easy observability, and exploiting surprising results.

Although these heuristics were identified in contexts involving much more complexity than the classic rule discovery tasks, there remains a question as to whether the heuristics found will generalize to even more complex situations. That is, as the task becomes more complex, will subjects use other strategies to deal with increased complexity in order

[1]This research was funded by a scholarship from la Formation de Chercheurs et l'Aide à la Recherche to the first author, and by grants from the National Institute of Child Health and Human Development (R01-HD25211) and the A.W. Mellon Foundation to the second author.

to make the task more tractable? This paper reports a study designed to investigate how subjects discover a complex concept. The analysis will focus on the strategies used by subjects to deal with difficulties encountered in discovering complex concepts.

Method

Overview. Subjects were shown the function of all but one command of a device. The subjects designed, conducted and evaluated experiments with the device to discover how the mystery command works. The mystery command was a complex sort operator that took three arguments.

Subjects. Twenty-two Carnegie Mellon University undergraduates took part in the experiment for course credit and eight dollars. Twenty-one of the subjects had taken at least one programming course, and all had used a computer before.

The Computer Interface. Subjects worked in a complex microworld in which a "milk truck" could execute a sequence of actions associated with a dairy delivery route. At any of 6 different locations along its route, it could toot its horn, deliver milk or eggs, or receive money or empties. The route of the milk truck was programmed using the keypad to enter a sequence of action-location parirs (see figure 1). As subjects entered their programs, the steps were displayed on the screen in the program listing. After the route had been entered, subject pressed 'RUN' and the milk truck executed its route on the screen. The milk truck went to each location on the programmed route in the order that it was programmed. The milk truck stopped at the location, and the subjects were shown by way of animated icons what transpired at the location. Also, as the route was being completed, a trace listing displayed in program format what transpired during that run (see figure 1).

When the mystery command, δ (delta), was not used, the trace listing was identical to the program listing. However, the δ command could change the order of delivery, and the resultant trace would then be discrepant from the program listing. Subjects could also look over the program and trace listings of their old programs.

The effect of the δ command was to reorder the execution sequence of part of the program according to the values of its three arguments, a number (1-6), a triangle (white or black), and a Greek letter, (α or β). Table 1 describes the effects of the delta command. The first and second programs in figure 1 show the effects of the δ with white triangle and α, and with black triangle and β.

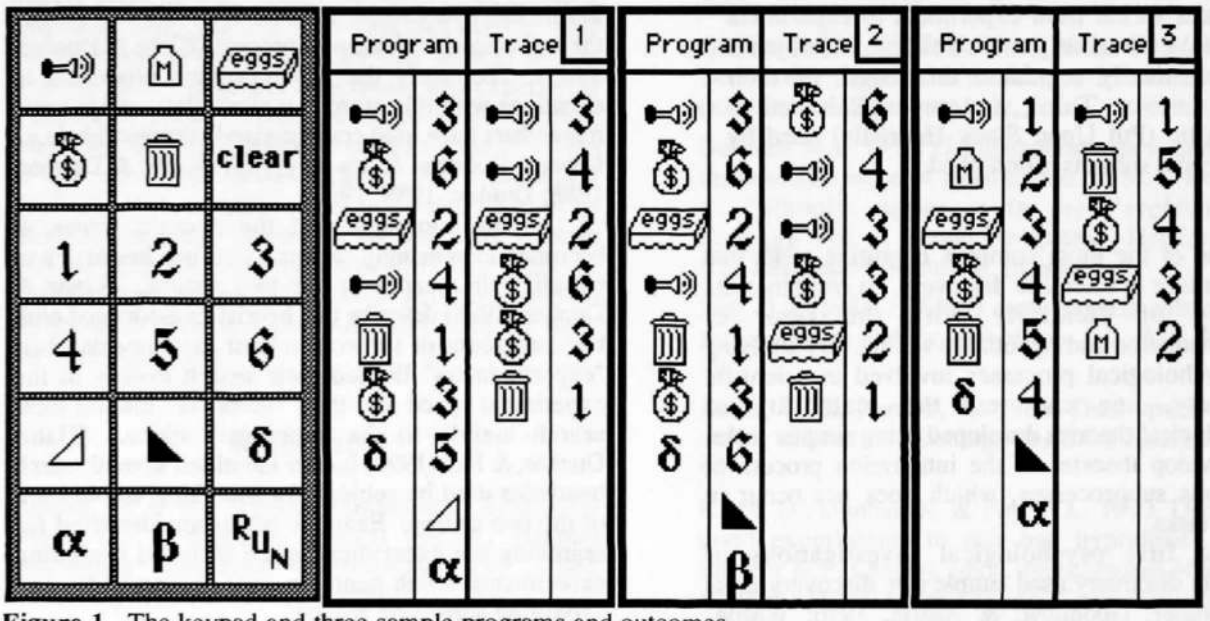

Figure 1. The keypad and three sample programs and outcomes.

For the last N steps in the program, δ reorders the execution sequence of the program by...

	White triangle (increasing)	Black triangle (decreasing)
α (item)	...items in keypad order.	...items in reverse keypad order.
β (number)	...increasing house number order.	...decreasing house number order.

Table 1. The function of the arguments to the δ command.

Materials. The interface was run on a Macintosh IIcx. Subjects were given a pen and scratch paper to take any notes that they wished during the task. A small audio tape-recorder and lapel microphone were used to record the verbal protocols.

Procedure. Subjects worked on the problem in two separate sessions on consecutive days. The first day consisted of an introduction to the task and 30 minutes of problem solving. The introduction to the task presented each aspect of the interface incrementally, and allowed the subjects to try a sample program before being introduced to the δ command. After being introduced to the δ command and its three arguments, subjects began experimentation with the explicit goal of discovering what the δ command and its three arguments did.

For the second session, subjects worked at the task until they had either solved it or had given up. If a subject falsely accepted an incorrect hypothesis, a counter-example was presented (see program 1 in figure 1). The same counter-example was used in all such cases. It was designed so that no subject would be likely to predict the actual outcome on the basis of an incorrect hypothesis.

Results

Overview. The 22 subjects produced 1103 total experiments over 33 hours. Analyses were conducted at three levels of detail: final solutions, computer protocols, and verbal protocols. The first two levels of analysis were carried out for all 22 subjects. Verbal protocols were carried out for only the first five subjects. These five subjects adequately represent the range of subjects: the second fastest solver, the second slowest solver, an average solver, a solver with counter-example, and a non-solver.

Subjects were grouped according to the success of their solutions. Three solution groups will be used for the analyses: subjects who solved without a counter-example (Solve); subjects who solved after receiving the counter-example (Challenge); and subjects who quit without solving (Quit). All subjects who received a counter-example eventually solved the task. The counter-example was usually given in the middle of the second day (mean program number 34.6, s=17.6). Table 2 presents the mean number of experiments and time on task for each group. The task was difficult but solvable: solution rates ranged from 30 to 179 minutes. Note that the Quit subjects did not simply give up prematurely: they ran slightly more experiments ($F(2,19)<1$) over a slightly longer period of time ($F(2,19)=2.6$, $p<.1$) than the other subjects (Quit vs. Solve Bonferonni/Dunn $t(19)=2.94$, $p<.05$).

Group	n	Experiments	Total time
Solve	11	49.5 (18.6)	78.6 (40.9)
Challenge	5	47.8 (17.2)	97.2 (34.2)
Quit	6	54.5 (17.6)	118.8 (17.8)
Total	22	50.4 (17.4)	93.8 (37.4)

Table 2. Mean number of experiments and time on task for each solution group (and standard deviation).

The MilkTruck domain was significantly more difficult for the subjects than the original BigTrak studies reported in Klahr & Dunbar (1988): the MilkTruck subjects ran 6 times as many programs over 2.5 times as much time.

Subjects varied a small number of experiment features. In the experiment space framework, an experiment can be viewed as a vector of features. One way to characterize a subject's search in the experiment space is to measure the number of features that change from one experiment to the next. In choosing how many features to vary, the subject must strike a balance between depth and breadth of search. If a subject varies few features, valid inferences can be made by comparing effects across programs. However, few feature changes between experiments may prevent the subject from reaching certain very informative experiments in finite time. On the other hand, if a subject varies many features, a broader range of experiments are tried at the cost of being able to make valid comparisons across experiments. These two main strategies have been described in concept attainment as "Conservative Focusing" and "Focus Gambling" (Bruner, Goodnow, & Austin, 1956).

For the following analyses four program features were used: the programmed route, and the three delta parameters. Since the subjects had access to more than just the last program, there are several ways of conducting the analyses, of which two have been chosen: comparing each program to the previous program, and comparing each program to the previous seven programs and choosing the lowest feature variation. Seven programs back represents the number of programs on the screen at all times. Comparison with the last program best represents the breadth of search. Comparison with the last seven programs is a measure of the validity of comparisons across experiments. To set a benchmark of comparison, the scores on these two measures by a completely random search in the experiment space

were computed using a Monte Carlo simulation[2]. The base rates for the random model were 2.49 features varied for comparing to the last program, and 1.45 features varied for comparing to the last seven programs. Subjects varied a mean of 1.77 features with respect to the last program, and 1.33 features with respect to the last seven programs. Both of these means were significantly lower than the random model base rates ($t(21)=-15.8$, $p<.0001$, and $t(21)=-3.52$, $p<.002$ respectively), suggesting that subjects preferred to do depth-first search. However, both measures were significantly greater than 1 ($t(21)=17.08$, $p<.0001$, and $t(21)=10.01$, $p<.0001$ respectively), indicating that subjects often chose programs which prevented them from being able to make valid comparisons with other programs.

Successful subjects varied features most effectively. To investigate the changes in the number of features varied across time, the two measures of feature changes were taken for each quartile of the total number of programs each subject wrote. An ANOVA was done on the subject means for each quartile for each solution group. Since the analyses of the two measures produced all the same effects, only the analyses of the measure of feature changes with respect to the last program are reported. Overall, the differences between the solution groups were not significant ($F(2,19)<1$). The quartile differences were significant ($F(3,57)=5.99$, $p<.003$). Post-hoc tests revealed that only the decrease from the first to second quartiles was significant (Scheffé $F(1,57)= 11.54$, $p<.001$). The interaction was marginally significant ($F(6,57)=1.86$, $p<.10$), with the Solvers tending to vary the same number of features across the quartiles, and the Challenge and Quit subjects showing a decrease in the number of features varied (See figure 2).

Since these data are correlational, the causal link between feature changes and solution group is ambiguous. However, one plausible account for the differences is that the early changing of many features in the Challenge and Quit subjects was one source of their problems. i.e., changing too many features at once produced ambiguous effects. For the differences in late features changes, the use of fewer feature changes in the Quit subjects was probably a symptom of their difficulties. i.e., they might be stuck investigating one particular situation. One would expect the Challenge subjects not to have this problem, since they all solved the task, and hence should look like the Solve subjects at the end of the task. Using the mean of the last five programs, the Challenge and Quit groups did differ significantly ($t(9)=2.17$, $p<.06$), with means of 1.88 and 1.43 feature changes respectively.

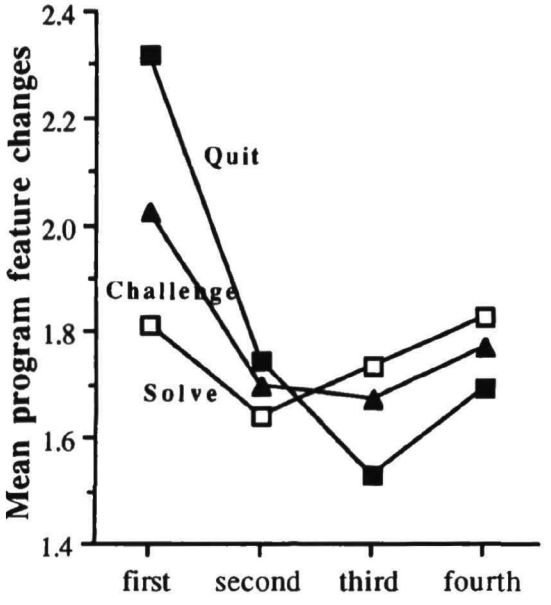

Figure 2. Mean number of program features varied with respect to the last program for each solution group for each quartile of total experiments.

The analyses of the changes in features emphasized the changes in δ parameters at the cost of treating all changes in the base program the same. However, subjects did some systematic variation of the basic program as well, as is shown in the following analysis.

Subjects systematically varied program length. Analyses of the program length revealed that most of the subjects used a strategy of gradually increasing the program length. When the length of each program was regressed against program number, 21 of the 22 subjects showed a positive slope, with a mean r of .56 ($\sigma=.22$). To further investigate the nature of this strategy, each subject's set of programs was divided into four equal parts, and an ANOVA of the means for each quartile for each solution group were calculated. The main effect for Solution group was not significant ($F(2,19)<1$). There were strong effects of quartile ($F(3,57)=9.4$, $p<.0001$). The quartiles increased in program length from the first to the fourth quartile, with significant increases between the first and second quartiles (Scheffé $F(3,57)=10.89$, $p<.05$), and the third and four quartiles (Scheffé $F(3,57)=7.43$, $p<.05$).

[2]A simulation of 100 random subjects was run having 50 programs each of mean length 5. For each program, a length was chosen randomly. For each step in the program, a delivery item and delivery location was chosen randomly. Similarly the delta parameters were chosen randomly. The random choices were all based upon a uniform distribution.

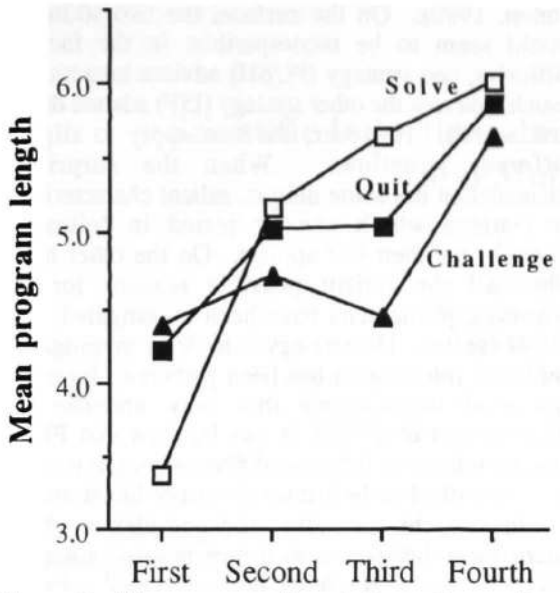

Figure 3. Mean program length for each quartile of the task for each solution group.

The interaction with Solution group was nonsignificant ($F(6,57)=1.41$, $p<.23$). As can be seen in figure 3, it is not simply the case that Solve subject ran longer programs which contained more information—although the differences were not significant, Solve actually started with slightly shorter programs, and were more consistent in their gradual increase of program length.

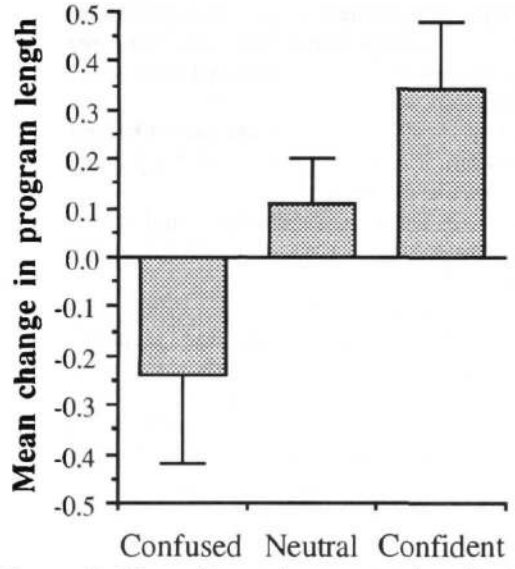

Figure 4 Mean change in program length at each confidence level (with standard error bars).

It is possible that program length could also be related to confidence levels. Since program length increases during the task, and presumably subjects are becoming more confident later in the task, one would expect that subjects were trying longer, more complex programs when they were confident. To test for such a relationship, the post program outcome statements from the full verbal protocols were coded for confidence levels on a 3 point scale: confused, neutral, and confident. The rater was blind to the actual program being run. An ANOVA was done on the change in program length from one program to the next for each confidence level (See figure 4). The effect of confidence level was significant ($F(2,255)=4.05$, $p<.02$), with post-hoc tests revealing a significant difference between the Confused and Confident means ($F(1,255)= 8.06$, $p<.02$).

Subjects used an explicit search strategy. When confronted with great difficulty while investigating a particular situation, subjects explicitly switched to investigating a different situation with the goal of returning to the confusing situation later. Upon completing their investigations of the new situation, subjects returned directly to the old situation, rather than a third, new situation. The time spent away from the original problem proved useful as subjects successfully then solved the old situation as well. This heuristic of putting a problem on hold and returning to it as soon as a different problem is solved was labelled the Put Upon Stack Heuristic (PUSH).

Of the five protocol subjects, only the three Solve subjects displayed evidence of using PUSH. Each of these subjects made explicit comments about being confused by the particular situation that they were currently investigating, and wanting to return to that situation later. And this is, in fact, what each of them did.

PUSH can be useful in three different ways. First, by enabling the subject to work on a different problem, PUSH allows new ideas to become activated and the activation of old ideas to decay, thereby reducing set effects and affecting the hypothesis space search in the old situation.

Second, the investigation of a different problem can suggest new operators which may be applied to the old situation, thereby improving the experiment space search. One subject's protocol provided evidence for this use of PUSH. This subject indicated that he was going to use PUSH: "Yeah. I'm just baffled right now. Ok, let's take a different tack. See if we can figure something out from a different angle. Let's go at it, looking at these other, what this other triangle does." Then, for the first time, he ran two programs which directly contrasted α with β, i.e., varied only α/β between two programs. This new operator proved successful in producing useful information to the subject, and lead the subject to say: "Let's try the same combination with other triangle again. I have a feeling that this might be what I need to be doing all along. The white triangle.

We'll start with α." The subject then went on using this operator, among others, to successfully solve the task.

Third, in inducing a complex concept involving an interaction such as the one used in the current experiment, discoveries about one part of the concept facilitate discoveries about another part of the concept. Thus, as the predictive power of hypotheses improve, the easier it is to resolve the remaining ambiguities. One subject seemed to use PUSH in this way.

Discussion

This experiment has demonstrated three strategies used by subjects in the induction of a complex concept. First, subjects tended to vary one or two features at a time between programs, with those varying the fewest features early more likely to solve. For strictly valid comparisons, scientific methodology states that only one feature may be varied at a time. If one were to take this strict view, then few subjects would be said to have used the strategy since the mean feature change for most subjects was well above one. However, there are two reasons why the number of feature changes in between program comparisons actually used by subjects may be lower than the simple numbers reported here: 1) the structure of the changes within a single program provides a large portion of the information; 2) subjects can design a sequence of programs which do not have to follow from the previous sequence.

Second, subjects start with short programs and gradually increased their length. Longer programs tend to be more informative, since they usually produce more step changes than short programs, and distinguish between more hypotheses. For example, a two step program has only two possible outcomes: the same order, or the reverse order. Thus, one would expect the solvers to have used longer programs. However, the use of long programs involves a high cognitive load in program design, interpretation, and memory. Thus, it is not necessarily better to always use longer programs. Indeed, the fastest solver used programs of a mean length of only 2.6 steps (1.9 standard deviations below the mean), indicating that long programs are not necessary for solution. Therefore, it is reasonable for subjects to start with short programs when they are relatively unfamiliar with the task, and use longer programs later in the task, when they need to test subtle differences between competing hypotheses. In fact, subjects did use shorter programs when they were less confident.

Third, subjects successfully made use of PUSH. Previous research has identified the Investigate Surprising Phenomena (ISP) strategy (Kulkarni & Simon, 1990). On the surface, the two strategies would seem to be incompatible: in the face of difficulty, one strategy (PUSH) advises breadth-first search, whereas the other strategy (ISP) advises depth-first search. However, the two apply to slightly different situations. When the surprising phenomenon has some unique, salient characteristics or features which can be tested in follow-up experiments, then ISP applies. On the other hand, when all the salient possible reasons for the surprising phenomena have been investigated, then PUSH applies. The strategy is to delay investigation until new information has been gathered. From the described mechanisms that may underlie the effectiveness of PUSH, it can be seen that PUSH may be related to incubation phenomena, especially those described in the history of science literature.

In sum, by extending the complexity of the microworld domain, several new heuristics used in scientific discovery have been revealed. Future studies need to address the applicability of these heuristics in other contexts, as well as assess the effects of the increased domain complexity on the use and effectiveness of previously described discovery heuristics.

References

Bruner, J. S., Goodnow, J. J. & Austin, G. A. (1956). *A study of thinking*. New York: Wiley.

Dunbar, K. (1989). Scientific reasoning strategies in a simulated Molecular genetics environment. In *the proceedings of the 11th annual meeting of the Cognitive Science society*, MI Ann Arbor, 426-433.

Dunbar, K. (1992). Concept Discovery in a scientific domain. Manuscript submitted for publication to *Cognitive Science*.

Klahr, D. & Dunbar, K. (1988). Dual space search during scientific reasoning. *Cognitive Science*, **12**, 1-48.

Klahr, D., Dunbar, K., & Fay, A. (1990). Designing Good Experiments to Test "Bad" Hypotheses. In J. Shrager & P. Langley (Eds.), *Computational Models of Discovery and Theory Formation*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Kulkarni, D. & Simon, H.A. (1988). The process of Scientific Discovery: The strategy of Experimentation. *Cognitive Science*, 12, 139-176.

Simon, H. A., & Lea, G. (1974). Problem solving and rule induction: A unified view. In L. W. Gregg (Ed.), *Knowledge and cognition*. Hillsdale, NJ: Lawrence Erlbaum Associates.