DESIGNING GOOD EXPERIMENTS
TO TEST BAD HYPOTHESES

Technical Report AIP - 83

David Klahr, Carnegie Mellon University
Kevin Dunbar, McGill University
Anne L. Fay, Carnegie Mellon University

15 August 1989

# The Artificial Intelligence and Psychology Project

Departments of
Computer Science and Psychology
Carnegie Mellon University

Learning Research and Development Center
University of Pittsburgh

(2)

# DESIGNING GOOD EXPERIMENTS TO TEST BAD HYPOTHESES

Technical Report AIP - 83

David Klahr, Carnegie Mellon University
Kevin Dunbar, McGill University
Anne L. Fay, Carnegie Mellon University

15 August 1989



To appear in Shrager, J. & Langley, P. (Eds.), *Computational Models of Discovery and Theory Formation.* Lawrence Erlbaum Associates (forthcoming).

This research was supported in part by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research, under Contract No. N00014-86K-0349, and in part by grant number OGP0037356 from the National Sciences and Engineering Research Council of Canada, and also in part by the Computer Sciences Division, Office of Naval Research, under contract number N00014-86-K-0678. Reproduction in whole or in part is permitted for any purpose of the United States government. Approved for public release; distribution unlimited.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

# REPORT DOCUMENTATION PAGE

| | |
|---|---|
| 1a. REPORT SECURITY CLASSIFICATION<br>Unclassified | 1b. RESTRICTIVE MARKINGS |
| 2a. SECURITY CLASSIFICATION AUTHORITY | 3 DISTRIBUTION / AVAILABILITY OF REPORT<br>Approved for public release;<br>Distribution unlimited |
| 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE | |
| 4. PERFORMING ORGANIZATION REPORT NUMBER(S)<br>AIP - 83 | 5. MONITORING ORGANIZATION REPORT NUMBER(S) |
| 6a. NAME OF PERFORMING ORGANIZATION<br>Carnegie-Mellon University<br>6b. OFFICE SYMBOL (If applicable) | 7a. NAME OF MONITORING ORGANIZATION<br>Computer Sciences Division<br>Office of Naval Research |
| 6c. ADDRESS (City, State, and ZIP Code)<br>Department of Psychology<br>Pittsburgh, Pennsylvania 15213 | 7b. ADDRESS (City, State, and ZIP Code)<br>800 N. Quincy Street<br>Arlington, Virginia 22217-5000 |
| 8a. NAME OF FUNDING / SPONSORING ORGANIZATION<br>Same as Monitoring Organizatior<br>8b. OFFICE SYMBOL (If applicable) | 9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER<br>N00014-86-K-0678 |
| 8c. ADDRESS (City, State, and ZIP Code) | 10 SOURCE OF FUNDING NUMBERS p4000ub201/7-4-86<br>PROGRAM ELEMENT NO. N/A; PROJECT NO. N/A; TASK NO. N/A; WORK UNIT ACCESSION NO N/A |

11. TITLE (Include Security Classification)

Designing Good Experiments to test Bad Hypotheses

12. PERSONAL AUTHOR(S)

Klahr, David, Dunbar, Kevin, & Fay, Anne L.

| 13a. TYPE OF REPORT | 13b. TIME COVERED | 14. DATE OF REPORT (Year, Month, Day) | 15. PAGE COUNT |
|---|---|---|---|
| Technical | FROM 86Sept15 TO 91Sept14 | 1989 August 4 | 36 |

16 SUPPLEMENTARY NOTATION

To appear in Shrager, J. & Langley, P. (Eds.), Computational models of discovery and theory formation. Hillsdale, NJ: Erlbaum (forthcoming)

| 17 COSATI CODES | | |
|---|---|---|
| FIELD | GROUP | SUB-GROUP |
| | | |
| | | |

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

scientific reasoning cognitie development
discovery processes
problem space search

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

SEE REVERSE SIDE

| | |
|---|---|
| 20. DISTRIBUTION / AVAILABILITY OF ABSTRACT<br>☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS | 21 ABSTRACT SECURITY CLASSIFICATION |
| 22a NAME OF RESPONSIBLE INDIVIDUAL<br>Dr. Alan L. Meyrowitz | 22b. TELEPHONE (Include Area Code) (202) 696-4302<br>22c. OFFICE SYMBOL N00014 |

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

SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

# Designing Good Experiments to Test Bad Hypotheses

**David Klahr, Kevin Dunbar, and Anne Fay. To appear in Shrager, J. & Langley, P. (Eds.), *Computational Models of Discovery and Theory Formation.* Erlbaum Associates (forthcoming).**

ABSTRACT

What does it take to design a good experiment? Given an hypothesis to be evaluated -- either in isolation or in competition with alternatives -- what formal rules, heuristics, and pragmatic constraints combine to yield a potentially informative experiment? How do subjects' expectations about the plausibility of an hypothesis effect the kind of experiments that they design, their ability to accurately observe and encode experimental outcomes and their responses to information that is consistent or inconsistent with the hypothesis? In this paper, we address these questions by creating a simulated discovery context and examining how subjects go about designing experiments to test hypotheses that are always, at the outset, incorrect. Thirtysix adult subjects were trained on the basic functions of a programmable robot. Then they were presented with a new function key (a "repeat" key) and asked to find out how it worked.

Subjects were given an initial hypothesis to test. They were told to write three "good experiments" to see if the key really worked as suggested, or if it did not, then to find out how it did work. *The Given hypothesis was always incorrect.* In some cases, it was only incorrect in a minor way, and in others it was from a different "frame" in which the meaning of several aspects of the hypotheses had to be reformulated in order to discover how "repeat" really worked. Given and Actual hypotheses also varied in how plausibl e they were.

Our results show that subjects are remarkably adept at designing and interpreting experiments in a novel domain. When subjects are given a plausible hypothesis, they tend to design an experiment that demonstrates the effect that is to be expected. When given inplausible hypotheses, they write programs that are good discriminators. When the discrepancy between the Given and the Actual hypothesis is very great, subjects are conservative in moving from one experiment to the next. When the Given is not very discrepant from the Actual, subjects are more likely to write experimental sequences that differ along several dimensions simultaneously. The challenge for builders of computational models of the experimental design process is how to capture the process whereby subjects bring to bear their general heuristics for "good experiments" in a novel domain.

## 1 Introduction

We share with the other contributors to this volume the ultimate goal of producing a computational model of the scientific discovery process, but we have approached this goal along a different path than most of them. Most of the chapters in this volume reflect a research strategy involving the creation of synthetic models, based on formal analysis of the demands of the discovery process, that result in running computer simulations. Empirical constraints on such models tend to derive from casual introspection, rather than extensive analyses of human performance.

A different, but complimentary approach to understanding the discovery process involves detailed analysis of the behavior of humans actually engaged in scientific discovery. This approach can be further divided into two paths. One path involves analyses of the scientific record of real scientists making real scientific discoveries (e.g., Darden, 1987; Langley, et al., 1987; Kulkarni & Simon, 1988). This path is necessarily coarse-grained, because the mental processes of the scientists must be inferred from either retrospective reports or laboratory notebooks. However, the face validity of such a data base is extremely high, because it has been deliberately selected as a consequence of having produced real scientific discoveries. The other empirically-constrained path -- and the one we follow in this chapter -- involves the creation (by the analyst) of simulated contexts for scientific discovery, and detailed analysis of moment to moment behavior of people -- typically ordinary college students -- operating in this context (e.g., Mynatt, Doherty & Tweeney, 1977). The disadvantage of this approach is that the discovery task itself is only analogous to science, rather than real science. However, its advantage is that it enables us to precisely control the context of the discovery process and to obtain fine-grained observations of the thinking processes surrounding that discovery.[1]

Scientific discovery can be characterized as a process involving search in two primary problem spaces -- a space of hypotheses and a space of experiments -- with additional searches of subsidiary problem spaces, including an observation space, an instrumentation space, a data analysis space, and a prior literature space (Newell, 1989). In our previous work (Klahr and Dunbar, 1988), we proposed a general framework (called "SDDS" for "Scientific Discovery as Dual Search") for the processes that coordinate and implement dual search in the experiment and hypothesis spaces. The framework was based on our empirical observations of subjects' behavior as they formulated hypotheses and designed experiments to evaluate them. In those studies, subjects were unconstrained with respect to both hypotheses and experiments. In this chapter, we narrow our focus to ask how people search the experiment space, when provided with a particular hypothesis to evaluate. By controlling both the hypothesis being evaluated and the extent to which it is correct, we are able to examine this search process in detail.

## 2 Designing experiments

What does it take to design a good experiment? Given an hypothesis to be evaluated - either on its own merits, or in competition with alternative hypotheses -- what formal rules, heuristics, and pragmatic constraints combine to yield a potentially informative experiment? How do subjects' expectations about the likelihood of an hypothesis being true or false affect the kinds of experiment that they design and their responses to information that is consistent or inconsistent with the hypothesis? These are the questions that we address in this chapter.

---

[1] There is an interesting third alternative that combines the features of each approach. It involves the presentation of the essential knowledge context faced by major scientists at the time of their discoveries to "ordinary", but technically trained, subjects, in order to see if they can make the same discovery, given the same information and the same goals as the original discoverer (Dunbar, 1989; Qin and Simon, 1989). In the few instances in which this has been done, some subjects were able to rediscover important scientific laws.

## 2.1 Previous studies

In our earlier studies (Dunbar and Klahr, 1989; Klahr and Dunbar, 1988) we instructed subjects about all of the basic features of a programmable robot, and then asked them to extend that knowledge by experimentation. This training provided a rich context that was intended to be analogous to a scientists' partial knowledge about a domain in which further information can only be obtained by experimentation. Our analyses focussed on subjects' attempts to discover how a new function operates -- that is, to extend their understanding about how the device works by formulating hypotheses and then designing experiments to evaluate those hypotheses; the cycle ultimately terminated when they believed that they had discovered how to predict and control the behavior of the device. In order to provide the substantive background for the studies to be reported in this chapter, we start by summarizing one of our earlier studies.

### 2.1.1 The BigTrak device

We used a computer-controlled robot tank (called "BigTrak") that is programmed using a LOGO-like language[2]. The device is operated by pressing various command keys on a keypad. BigTrak is programmed by first clearing the memory with the CLR key and then entering a series of up to sixteen instructions, each consisting of one of its eight function keys (the command) and a 1- or 2-digit number (the argument). When the GO key is pressed BigTrak then executes the program. To illustrate, one might press the following series of keys: **CLR ↑ 5 #←# 7 ↑ 3 #→# 15 HOLD 50 FIRE 2 ↓ 8 GO.** BigTrak would then do the following: move forward five feet, rotate counterclockwise 42 degrees (corresponding to 7 minutes on an ordinary clock face), move forward 3 feet, rotate clockwise 90 degrees, pause for 5 seconds, fire twice, and backup eight feet.

### 2.1.2 General method of previous work

First, we established a common knowledge base about the device for all subjects, *prior* to the discovery phase. We instructed subjects about how to use each of the basic function keys. Then the discovery phase started. Subjects were told that there is a "repeat" key, that it takes a numerical parameter, and that there can be only one RPT in a program. They were asked to discover how RPT works by proposing hypotheses and running programs with RPT in them in order to test their hypotheses. (On the original BigTrak, there was only one way that RPT worked: it repeated the previous N instructions once. In the studies reported here, we used several different rules for how RPT works). Subjects generated a concurrent verbal protocol that included hypotheses, experiments (programs), observations, evaluations, and revised hypotheses.

### 2.1.3 Results

Nineteen of the 20 adult subjects in our first study (Klahr and Dunbar, 1988) discovered how the RPT key works within the allotted 45 minutes. The mean time to solution was 19.8 minutes. Subjects generated, on average, 18.2 programs.

Protocols were encoded in terms of the hypotheses listed in Table 1. We defined a "common hypothesis" as a fully- specified hypothesis that was proposed by at least two different subjects. Across all subjects, there were 8 distinct common hypotheses. Subjects did not always express their hypotheses in exactly this form, but there was usually little ambiguity about what the current hypothesis was. We coded each experiment in terms of the hypothesis held by the subject at the time of the experiment, and Table 1 shows the proportion of all experiments that were run while an hypothesis was held. (As noted earlier, HS1 in Table 1 is the way that BigTrak actually operated.)

Subjects proposed, on average, 4.6 different hypotheses (including the correct one). Fifty-five percent of the experiments were conducted under one of the eight common hypotheses listed in Table 1. Partially-specified hypotheses, which account for 3% of the experiments, were defined

---

[2]This device was first used by Shrager (1985) in his investigation of "instructionless learning" (Shrager & Klahr, 1986)

**Table 1:** Common hypotheses and percentage of experiments conducted under each. Hypotheses are labeled according to the role of N: HS - selector; HN - nil; HC - counter.

| HYPOTHESIS | CURRENT DESIGNATION[3] | % EXPERIMENTS UNDER EACH HYPOTHESIS |
|---|---|---|
| **HS1: One repeat of last N instructions.** | D | 02 |
| HS2: One repeat of first N instructions. | | 04 |
| **HS3: One repeat of the Nth instruction.** | C | 03 |
| HN1: One repeat of entire program. | | 06 |
| HN2: One repeat of the last instruction. | | 04 |
| **HC1: N repeats of entire program.** | A | 14 |
| **HC2: N repeats of the last instruction.** | B | 20 |
| HC3: N repeats of subsequent steps. | | 02 |
| Partially specified | | 03 |
| Idiosyncratic | | 14 |
| No Hypothesis | | 28 |

as those in which only some attributes of the common hypotheses were stated by the subject. (E.g., "It will repeat it N times.") An idiosyncratic hypothesis was defined as one that was generated by only one subject. Such hypotheses are not listed separately in Table 1. For 28% of the experiments, there were no stated hypotheses.

## 2.2 The hypothesis space

The eight common hypotheses -- which account for over half of the experiments -- can be represented in a space of "frames" (cf. Minsky, 1975). The basic frame for discovering how RPT works is depicted at the top of Figure 1. It consists of four slots, corresponding to four key attributes: (1) the role of N: does it *count* a number of repetitions or does it *select* some segment of the program to be repeated? (2) The unit of repetition: step, program or group of steps? (3) Number of repetitions: (1, N, some other function of N, or no role at all? (4) Boundaries of repeated segment: beginning of program, end of program, Nth step from beginning or end? A fully-instantiated frame corresponds to a fully-specified hypothesis, several of which are shown in Figure 1. There are two principle subsidiary frames for RPT, N-role:*counter* and N- role:*selector*. Within each of these frames, hypotheses differing along only a single attribute are shown with arrows between them. All other pairs of hypotheses differ by more than one attribute. Note that the hypotheses are clustered according to the *N*-role frame in which they fall. No arrows appear between hypotheses in one group and the other because a change in *N*-role requires a simultaneous change in more than one attribute. This is because the values of some attributes are linked to the values of others. For example, if *N*-role is *counter*, the number-of-repetitions is *N*, whereas, if *N*-role is *selector*, then number-of-repetitions is *1*.

---

Insert Figure 1 about here

---

This frame representation is a convenient way of capturing a number of aspects of the scientific reasoning process. First, it characterizes the relative importance that subjects give to different aspects of an hypothesis. Once a particular frame is constructed, the task becomes one of filling in or verifying "slots" in that frame. The current frame will determine the relevant attributes. That is, the choice of a particular role for *N* (e.g., *N-role:counter*) also determines what slots remain to

---

[3]Entries in this column show the labels for the four key hypotheses to be used in the present study. Note that they include the two most "popular" (A and B) and the two least popular (D and C).

be filled (e.g., number-of-repetitions:*N*), and it constrains the focus of experimentation. Furthermore, frames enable us to represent the differential importance of different attributes, as the "frame type" becomes the most important attribute, and its "slots" become subordinate attributes. This is consistent with Klayman and Ha's (1985) suggestion that "some features of a rule are naturally more 'salient', that is, more prone to occur to a hypothesis-tester as something to be considered" (p. 11). In our context, a frame is constructed according to those features of prior knowledge that are most strongly activated, such as knowledge about the device or linguistic knowledge about "repeat." When a frame is constructed, slot values are set to their default values. For example, having selected the *N*-role:*counter* frame, values for number-of- repetitions, units and boundary might be chosen so as to produce HC1 (see Figure 1).

## 2.3 The experiment space

Subjects tested their hypotheses by conducting experiments; i.e., by writing programs that included RPT and observing BigTrak's behavior. But it is not immediately obvious what constitutes a "good" or "informative" experiment. In constructing experiments, subjects are faced with a problem-solving task that parallels their effort to discover the correct hypothesis, except that in this case search is not in a space of hypotheses, but in a space of experiments.

A useful characterization of the experiment space is one that abstracts over the specific content of programs and refers to only two of their attributes. The first is $\lambda$ -- the length of the program preceding the RPT. The second is the value of N -- the argument that repeat takes. Within the $\lambda$ - N space, we identify three distinct regions according to the relative values of $\lambda$ and N and their limiting values.[4]The regions are depicted in Figure 2, together with illustrative programs. Region 1 includes all programs with RPT 1. Region 2 includes all programs in which the value of N is greater than 1 but less than $\lambda$. Region 3 includes all programs in which N is equal to or greater than $\lambda$.

---

Insert Figure 2 about here

---

Programs from different regions of the E-space vary widely in how effective they are in supporting or refuting different hypotheses. Figure 3 shows how BigTrak would behave under different rules when executing programs from different regions of the E-space. The programs are depicted on the left by generic commands (e.g., *X, Y, Z*) and the device behavior is shown under the column corresponding to each of the four rules used in this study. Specific program content is abstracted in this figure in two forms. In the upper part of the figure (Examples 1 - 8), the commands are presumed to be maximally differentiated, so that they are easily identified in the behavior of BigTrak. In the lower part of the figure (Examples 9 - 16), the same $\lambda$ - N programs are used, but with the same command in each position.

To illustrate, the second example shows a two-step program with N =1. This is a Region 1 experiment. Under Rule A, the two-step program would be executed once and then repeated one more time. Under Rule B, only the last step (y) would be repeated one additional time. Under rule C, the first step would be repeated once, and under Rule D, the last N steps (in this case, the last step, since N = 1) would be repeated once. This program cannot discriminate between Rules B and D.

It should be noted that the upper and lower examples in Figure 3 represent extremes of a continuum of discriminating power. The most highly differentiated programs are obtained if we substitute distinct commands for x, y and z (e.g., X = ↑ 2, Y = FIRE 1, etc.). The least informative

---

[4]In previous analyses (Klahr & Dunbar, 1988, Dunbar & Klahr, 1989), we used a finer-grained categorization of the Experiment Space into six regions. The mapping from the earlier to the current regions is as follows: I & II #→# 1, III #→# 2, IV & V & VI #→# 3. The $\lambda$=1, N=2 cell from the old Region I goes into the new Region 3.

programs are those in which both the command and the parameter are the same (e.g, each X = #→# 15 in examples 9 - 16.) Intermediate between these two extremes are programs in which the commands are the same, but the parameters are different, such as FIRE 1 FIRE 2 FIRE 3. For many such programs, behavior under the different rules is in fact distinct, but it is extremely difficult to keep track of BT's behavior. We will present one such example in Section 4.1.

---

Insert Figure 3 about here

---

Two important and subtle features of the rules are included in the notation in Figure 3. The first potentially confusing feature has to do with the ambiguity inherent in the phrase "repeat it N times". Does it mean N, or N+1 total executions of the repeated entity? That is, if a program is supposed to repeat something twice, a subject might expect to observe either two or three occurrences of that item or segment. The underlined segments in Figure 3 show the behavior generated by the N+1 interpretation (which is the one we use in our simulations). If subjects use the N interpretation, then they would not expect to see these extra segments. The second feature involves rules C and D when $N > \lambda$. For these programs (indicated by an asterisk), N is set equal to $\lambda$. Experiments in the 3 regions interact with hypotheses as follows:

1. Programs in Region 1 have poor discriminating power. Experiments in this region are typically generated by subjects who are attempting to demonstrate a counter hypothesis, and are simply choosing a minimal number of repetitions. We have already described example 2, and example 3 similarly fails to discriminate B from D. Example 1 -- a minimalist program with $\lambda = N = 1$ -- has no discriminating power whatsoever.

2. Region 2 provides maximal information about all of the most common hypotheses, because it can distinguish between counters and selectors, and it can distinguish *which* selector or counter is operative. It produces different behavior under all four rules for any program in the region, as long as the commands are reasonably distinctive. However, even this region fails to discriminate among some rules for some minimally distinctive programs (see Rules B and D in Example 12 in Figure 3.)

3. For Selector hypotheses, programs in Region 3 are somewhat confusing, because they are executed under the subtle additional rule than values of N greater than $\lambda$ are truncated to $N = \lambda$, and varying N in this region will give the impression that N has no effect on the behavior of the device. Although some of the programs in this region are discriminating (e.g., example 5, with $\lambda = N = 3$), others either don't discriminate at all (e.g., C vs D in example 7), or they depend on the truncation assumption to be fully understood (e.g., examples 6 - 8). A more serious problem with Region 3 is that a series of experiments that incremented N by 1 would, under rules C or D, lead to the conclusion that N had no effect (e.g., compare examples 1 and 7 or 5 and 8.)

## 3 An empirical study of testing an incorrect hypothesis

The brief summary of our earlier studies provides a context for the present paper in which we describe a new study designed to explore subjects' response to negative feedback. In these new studies, we always provide subjects with an initial hypothesis about how RPT might work. It is always wrong. In some conditions it is only "somewhat" wrong, in that it is from the same frame as the way that RPT really works. In others, it is "very" wrong: in that the suggested hypothesis comes from a different frame than the actual rule.

### 3.1 Subjects

Thirty-six Carnegie Mellon undergraduates (27 males and 9 females) participated in the experiment for course credit. Most were science or engineering majors. All of the subjects completed a questionnaire about their programming experience and skill, and a self-rating of their skill in math, science, and mechanical reasoning. Subjects reported having taken between 0 and 5 programming courses (mean 2.1, sd 1.4), and they tended to rate themselves between average and above average on all the technical and scientific scales.

### 3.2 Procedure

All subjects worked with a simulated version of the BigTrak on a Xerox Dandelion workstation. The workstation enabled us to control the way that RPT actually functioned, as well as facilitating the recording of the experiments that subjects wrote to evaluate their hypotheses.

There were three phases to the study. In the first, subjects were introduced to the (simulated) BigTrak (henceforth referred to as BT) and trained to criterion on all of its basic commands. In the second phase, subjects were told that there was a RPT key, that it required a numeric parameter, and that there could be only one RPT in a program. They were told that their task was to find out how RPT worked by writing programs to test a particular hypothesis. At this point, the Experimenter suggested one possible way that RPT might work, and instructed the subject as follows:

> "write down three good programs that will allow you to see if the repeat key really does work this way. Think carefully about your program and then write the program down on the sheet of paper .... Once you have written your program down, I will type it in for you and then I will run it. You can observe what happens, and then you can write down your next program. So you write down a program, then I will type it in, and then you will watch what the program does. I want you to write three programs in this way. (The complete script for this part of the procedure is shown in Appendix I.)

Next, the third phase began. Subjects wrote programs (experiments) to evaluate the given hypothesis. Although subjects did not have access to a record of the behavior of the device under earlier experiments, they did have access to the list of programs that they had written, and they often referred to them in commenting on differences between the most recent outcome and previous ones. Subjects were instructed to give verbal protocols. This gave us a record of: (a) what they thought about the kinds of programs they were writing while testing their hypotheses; (b) what they observed and inferred from the device's behavior; and (c) what their hypothesis was about how RPT actually worked. When subjects had written, run, and evaluated three experiments, they were given the option of writing additional experiments if they were still uncertain about how RPT worked.

### 3.3 Design

The BT simulator was programmed so that each subject worked with a RPT command obeying one of the four rules listed in Table 2. Note that there are two "counter" rules and two "selector" rules. Table 2 also summarizes the results from two earlier studies indicating that the counter hypotheses are very common, whereas the selector hypotheses are very rarely proposed by subjects. Thus, in our previous work counter hypotheses were regarded as highly probable, and selector hypotheses were regarded as improbable. Thus, we expected that the *apriori* belief in particular hypotheses would have a large effect on the types of experiments designed and the interpretation of results.

Another crucial part of the design of this study is that *RPT never worked in the way that was suggested.* The design is shown in Table 3. The Given hypothesis is the one that was suggested

**Table 2:** RPT Rules and Hypotheses used and Results from Earlier Studies.

| | | Proportion of Experiments Run under each Hypothesis | |
|---|---|---|---|
| | | Study 1 | Study 2 |
| Counters | | | |
| | A: Repeat the entire program N times | 14 | 13 |
| | B: Repeat the last step N times | 20 | 26 |
| Selectors | | | |
| | C: Repeat the Nth step once | 3 | 5 |
| | D: Repeat the last N steps once | 2 | 5 |

by the experimenter, and the Actual hypothesis is the way that BT was programmed to work.[5] We used a between-subjects design, with 3 subjects in each of the given- actual conditions ($N = 36$). This design yielded 12 subjects in within-frame conditions (e.g., Repeat the program N times (A) #→# Repeat the last step N times (B); Repeat last N steps (D) #→# Repeat Nth step once (C) ). There were 24 subjects in the between-frame conditions (e.g., Repeat the program N times (A) #→# Repeat last N steps (D); Repeat Nth step once (C) #→# Repeat the program N times (A)). To the extent that our assumptions about the frame-like nature of hypotheses is correct, we would expect the between-frame conditions to be more difficult than the within-frame conditions. We also expect subjects to have less difficulty discovering the (initially favored) counter hypotheses than the (non-favored) selector hypotheses.

**Table 3:** Design of given-actual conditions.

| | | ACTUAL | | | |
|---|---|---|---|---|---|
| | | Counter | | Selector | |
| Given | Counter | A #→# B | | A #→# C | A #→# D |
| | | B #→# A | | B #→# C | B #→# D |
| | Selector | C #→# A | C #→# B | C #→# D | |
| | | D #→# A | D #→# B | D #→# C | |

## 3.4 Questions about searching the experiment space.

Now that we have described the details of our design, we can pose the following specific questions:

1. With respect to overall effort and success rates:

   a. Will subjects have more difficulty when they have to change frames in order to discover the Actual rule than when they can remain within the same frame as the Given hypothesis?

   b. Will subjects find it easier to discover rules from the preferred frame (Counters) than from the non-preferred frame (Selectors)?

---

[5] In our discussion, we will distinguish among three categories of hypotheses and/or rules. *Given* hypotheses are the ones initially suggested by the experimenter, *Active* hypotheses are the ones currently being evaluated by the subject, and *Actual* rules are the ways that RPT actually works in a particular condition. Ideally, a subject would start with Active = Given and end with Active = Actual.

c. Will the difficulty of crossing frame boundaries interact with the preference for some hypotheses rather than others? That is, will it be easier to discover a counter when given a Selector hypothesis, than to discover a selector given a counter hypothesis?

d. Will the extent of "rational" search of the hypothesis space depend on experimental conditions? To what extent will subjects propose hypotheses that are consistent with the evidence available to them?

e. Will subjects find it easier to reject hypotheses that have been given to them, rather than hypotheses that they have generated themselves? In our prior research, most subjects began with hypotheses A and B, and needed a considerable amount of disconfirming evidence before abandoning their hypotheses. When subjects are actually given hypotheses to test they may more readily abandon their hypotheses.

2. With respect to search in the Experiment Space:

a. Will subjects' interpretation of what a "good experiment" is vary according to the Given-Actual condition in which they find themselves? That is, will experiments for favored hypotheses tend to demonstrate the presumed effect of RPT, while experiments for unfavored hypothesis tend to have the power to discriminate between alternative hypotheses?

b. To what extent will subjects adopt the same Experiment Space as we have presented here? Will their choice of experiments reflect an implicit understanding of the interactions shown in Figure 3?

c. What kinds of pragmatic rules will subjects apply to their search of the E-space? Will they design programs that are easily observable, discriminatory and memorable?

3. With respect to the observation and encoding of experimental outcomes:

a. Given that the Big Trak never works the same way as the Given hypothesis, how will subjects interpret the disconfirming evidence?

b. Will disconfirmation result in subjects searching a new region of the experiment space?

c. Will hypothesis preference also influence subjects' encoding and evaluation of experimental outcomes as well as overall success rates? That is, will subjects tend to distort their encoding of evidence in the direction of confirming favored hypotheses?

## 4 Results

The raw data are comprised of subjects' written programs as well as transcriptions of subjects' protocols (i.e., verbalizations) during the experimental phase. The protocols provided the basis for all of our measures of hypotheses changes and search in the experiment space. In Section 4.1, we informally describe two characteristic protocols, and then in subsequent sections, we provide a quantitative analysis based on the full set of protocols.

## 4.1 Complete Protocols

The subject protocols are extremely rich, and in this section, our aim is only to convey a general sense of the kind of encodings and inferences that we make from them. In the following two summaries, we focus on the ease with which subjects coordinate their search in the Hypothesis and Experiment spaces. The complete protocols are listed in Appendix A and Appendix B. Line numbers correspond roughly to major clauses. For each experiment, the commands used in the program are listed on the left side of the Table and the actual behavior of BigTrak is shown in boldface type on the right side. Experimenter comments are shown in uppercase.

### 4.1.1 Subject DP

Subject DP had experience with several programming languages (LOGO, LISP, Pascal) and reported to have had between 100 to 500 hours of programming experience. He rated himself as "above average" in math and science, and average in "handling new gadgets." DP was in the Counter #→# Selector condition, he was given Rule A: *Repeat entire program N times.* The actual rule was Rule C: *Repeat Nth step once.* DP discovered the correct rule after 5 experiments.

Several general characteristics of DP's protocol make it interesting (but not unusual). First, even before the first experiment, DP rejects the given hypothesis and proposes an alternative (003: "I want to test to see if repeat repeats the statement before it", e.g., this is rule B, not rule A.) Second, throughout the experimental phase, DP makes many explicit comments about the attributes of the experiment space. He clearly attends to the properties of a "good" experiment. Third, DP operates in an experiment space that includes a feature that we have ignored so far: whether the range of influence of RPT extends to commands that precede it, follow it, or both. (We have included only the first of these in our analysis so far.) Several of our subjects explore this possibility, but it was not a dominant focus for most experiments.

DP first focuses on the question of the before/after range of RPT, and he writes a minimal program with one step on each side of RPT. Note that he uses easily discriminated commands (left and right turns) so that if RPT is having an effect on either side of its location in the program it will be unambiguously evident. (This ability to write programs that contain useful "markers" is an important feature of our subjects' behavior, and we will return to it later). DP is very clear about his intentions in his first experiment (003-010): to determine whether RPT acts on instructions before or after the RPT command. To resolve this question DP conducts an experiment with commands both before and after the RPT key. This experiment is appropriate as it allows DP to discriminate between these two rival hypotheses. . However, with respect to being able to discriminate between the Given hypothesis (A), the Active hypothesis (B) and the Actual hypothesis (C), the program will yield ambiguous results. DP extracts from the first experiment the information he sought (017: "it appears that the repeat doesn't have any effect on any statements that come after it.")

For the second experiment DP returns to the question of whether the Given hypothesis (A), or the Active hypothesis (B) is correct, and he decides to increase $\lambda$ from 1 to 2. He also decides to include one step following the RPT "just to check" that RPT has no effect on instructions that follow it (022-023). Thus, DP is in fact testing 3 hypotheses; A, B, and 'after'. Once again, he uses commands that can be easily discriminated. He continues to write a program from Region 3 of the E-space ($\lambda$=2, N=2). DP observes that there were 2 executions of the $\uparrow$ 2 instruction, and he concludes (028) that "it only repeats the statement immediately in front of it." While this conclusion is consistent with the data that DP has collected so far, the hypothesis (B) is not in fact how the RPT key works.

For the third experiment, DP continues to put commands after RPT just to be sure they are not affected. However, given that his active hypothesis has been confirmed in the previous experiment he now decides to write a program that further increases the length of the program. This is his first experiment in Region 2. The goal of this experiment was to "see what statements are repeated" (032). He realizes that the outcome of this experiment is inconsistent with his

Active hypothesis (B), while the outcome of the previous experiment was consistent with B (050: "... it seemed to act differently in number two and number 3"). The unexpected result leads DP to abandon Hypothesis B, and he decides to continue beyond the mandatory three experiments.

For the fourth experiment, DP uses a different value of N (055: "just to see if that [a value of 3 instead of 2] has anything to do with it.") Here too, DP demonstrates another important characteristic of many of our subjects' general approach to experimentation. He uses a very conservative incremental strategy, similar to the VOTAT (vary one thing at a time) experimental strategies described by Tschirgi(1980) and the Conservative Focusing strategy described by Bruner, Goodnow, & Austin (1956). This approach still leads him to put commands after the RPT, even though he seems confident that RPT has no effect on them, and even though they place greater demands on his observational and recall processes. (At the λ - N level, DP executes VOTAT consistently throughout his series of five experiments. The λ - N pairs are: 1-2, 2-2, 3-2, 3-3, 3-1. For the last three experiments, even the specific commands and their parameters remain the same, and only N varies.) This moves him from region 2 into region 3, and while analyzing the results of this experiment (061 - 071) in conjunction with earlier results, DP changes from the counter frame to the selector frame. First he notices that "the number three" statement (i.e., the ↓ 1) was repeated twice in this case but that "the turning statement" was repeated (i.e., executed) only once (061 - 063). The implied comparison is with the previous experiment in which the turning statement (i.e., "the right 15 command" [064]) was the command that got repeated. The next sentence is of particular interest:

> .... because when I change the number not only did it change ... "it didn't change the uh .... the number that it repeated but it changed the uh.... the actual instruction" (066 - 069).

We believe that DP is attempting to articulate a change from the Counter frame to the Selector frame, as the following paraphrase of his comments indicates:

> When I changed the value of N, it didn't change the *number* of repetitions, but it did change *which* commands got repeated.

DP goes on to clearly state two instantiated versions of the correct rule by referring to previous results with N = 2 and N = 3, and he designs his fifth experiment to test his prediction with N =1. The outcome of this final experiment, from Region 1, in conjunction with earlier results is sufficient to convince him that he knows how RPT works.

### 4.1.2 Subject JS

JS also rated himself as above average in math and science; as well as in "handling new gadgets." He reported having had between 50-100 hours of programming experience. This subject was also in the A #→# C condition. JS's protocol has two interesting features. First, he never fully accepts the Given hypotheses (A: Repeat entire program N times), and at the very outset, he proposes a few alternatives. Second, he is very articulate about several aspects of his experimental strategy, not only with respect to both the λ - N space, but also in terms of the logic of a disconfirmatory strategy, as well as pragmatic constraints, such as designing programs that are easy to observe and encode.

JS starts by expressing doubt about the Given hypothesis and setting out to disconfirm it (002-005), while using a "simple program" (006) with "distinct steps" (009) that can be "distinguished" (012). As he develops the program, he proposes two alternative hypotheses, and reasons about them on the basis of plausibility and functionality (013 - 017). As he develops his first program, JS describes its predicted behavior as if his Active hypothesis was Repeat Nth step once, which is the Actual rule. That is, he expects the RPT 1 to execute the ↑ 1 after the ↓ 1 which will "bring it back to its original position" (022). JS also adds a command following the RPT just to see if RPT has any effect on subsequent commands, although he doesn't seem to expect it to.

The Experimenter now asks JS to make a prediction before running the program (032), and JS gives two possible outcomes. If the Given hypothesis is correct, then he predicts that after the

program is executed the first time, that it will be executed again in its entirety: "it will continue with the rest of the program" (037). However, if his alternative hypothesis (C) is correct, then "the only thing I'm thinking it might do is I think it might just move forward 1 (i.e., repeat the first step only) and then it'll end up turning to the left 30" (038 - 040).

The program runs, and JS correctly observes and interprets its behavior as disconfirming the Given, but confirming his Active (and the Actual) hypothesis (042 - 049). However, JS then realizes that a Region 1 program does not rule out another plausible hypothesis: Repeat first N steps once. He deliberates for a bit on what kind of experiment would best discriminate between the two possibilities, and for his second experiment he constructs a Region 2 program with $\lambda = 4$ and $N = 3$, and four highly discriminable commands. He also articulates a VOTAT strategy (061 - 063: "I want to run the same program, because I know what it does, I just want to change the condition of the repeat.")

At this point JS states the correct rule as well as his now-disconfirmed hypothesis.

> So it's just repeating the step number of the ... the number you put after the repeat it repeats that sequence, ... it doesn't repeat first second and third like I thought it might, it just repeats the third step. (077-082)

Having discovered the correct rule, JS goes on to explore the effect of having $N > \lambda$, and writes one more experiment to attempt to resolve that question. He appears to end somewhat unsure of this subtle feature.

#### 4.1.3 General features of subjects' behavior

We have presented only two of our 36 protocols, but they suffice to illustrate several general features of subjects' approach to this task.

1. Subjects often were unwilling to accept the Given hypotheses. Recall that both JS and DP expressed doubt about the Given hypothesis *prior* to running their first experiment and proposed an alternative. This initial skepticism, as we noted earlier, was not uncommon, and it varied in its degree and in the conditions under which it occurred. We can define "mild skepticism" as the consideration of an alternative hypothesis from the same frame as the Given hypothesis, and "extreme skepticism" as the consideration of an alternative from a different frame. Table 4 shows the frequency with which different types of hypotheses were entertained prior to the first experiment. For both Counter and Selector subjects, nearly two-thirds (17/26) of the hypotheses that were considered (over all 18 subjects in each condition) were the Given hypotheses. However, when we look at what *additional* hypotheses were generated by subjects in the two Given conditions, we see a strong effect of condition. In addition to the Given hypothesis, Counter subjects were most likely to propose the other Counter, or else an idiosyncratic hypothesis. They rarely suggested a Selector. In contrast, Selector subjects were highly likely to pose Counter alternatives. Put another way, while 78% of all non-Given hypotheses suggested by Selector subjects were Counters, only 22% of all non-Given hypotheses suggested by Counter subjects were Selectors. Extreme skepticism occurs mainly among subjects in the Given = Selector group. This suggests that the apriori strength of an hypothesis is the determining factor in whether an hypothesis from another frame is proposed prior to experimentation.

2. Most subjects showed a clear understanding of the two principle dimensions of the $\lambda$ - N space. Their protocols are filled with comments about "using longer programs", "using a different value of N", etc. At a finer grain of analysis, subjects were also aware of the importance of what might be called "good instrumentation" -- designing programs that have identifiable markers in them. We already saw one such example in Subject JS (Appendix B lines 007 - 009). The following statements by other subjects are typical: (emphasis added)

**Table 4:** Frequency of hypotheses actually tested on First Experiment.

| | Hypotheses tested on First experiment | | | |
|---|---|---|---|---|
| Given | Given | Counter | Selector | Other |
| COUNTER | 17 | 4 | 2 | 3 |
| SELECTOR | 17 | 7 | 1 | 1 |

> I don't want to have two of the same move in there yet, *I might not be able to tell if it was repeating the first one or if it was doing the next part of my sequence.* (AD03)

> I'm just going to make up some random but different directions *so that I'll know which ones get executed.* (RS22)

> I'm going to use a series of commands that will .... *that are easily distinguished from one another,* and won't run it off the screen. (GM27)

> ... so I'm going to pick two [commands] that are the direct opposite of each other, to see if ... they don't really have to be direct opposites but .. anyhow, I'm just going to write a program that consists of two steps, *that I could see easily.* (BB04)

In addition to working in both the $\lambda$ - N space, and the instrumentation space, subjects were generally sensitive to pragmatic constraints such as using small values of N on commands so that BT's behavior could be easily observed and remembered.[6]

3. Although many subjects could articulate these general strategies, they could not always carry them out, as the following selection from subject MA indicates. MA was in the most difficult Selector #→# Counter condition. He was given D: Repeat last N steps, and the Actual rule was B: Repeat last step N times. MA expresses some doubt about the Given hypothesis, and articulates a good experimental strategy:

> Ok, if it repeats the last N steps --- which we are presuming: it may or may not do that --- if it does then you'd want to write a program which would have a certain amount of steps before the repeat key, and not repeat all of them, so that you could see if it actually does that. I'm also going to add steps after it so that if it repeats the steps after it you'll be able to see that. So the problem is just making up steps that you can differentiate between, so that's what I'm going to do.

Unfortunately, he decides to write a program that includes *only* FIRE commands, and although he expresses some doubt about whether he will be able to interpret its behavior, he proceeds as planned:

> I'll have it, ok well I was thinking of having it not move anywhere, just fire, but I don't know if I'll be able to tell those apart on the screen. So why don't I do that anyway? So I'm just going to fire once, I'll fire twice, I'll fire 3 times and then I'll repeat the previous 2 steps, then I'll have it fire four times, then fire 5 times. **[FIRE 1, FIRE 2, FIRE 3, REPEAT 2, FIRE 4, FIRE 5]**

---

[6] Although it is not shown in this paper, such awareness of pragmatic constraints contrasts markedly with the behavior of middle-school children in the same situation (Dunbar, Klahr and Fay, 1989).

At this point BigTrak fires 21 times, but MA counts 23 FIREs, gets confused, and abandons the all-FIRE approach to experimentation.[7]

4. In addition to these general characteristics of individual programs, many subjects were systematic in the *sequence* of programs that they wrote, following, as suggested earlier, a strategy of varying only one thing at a time (i.e., changing either λ or N, but not both from one experiment to the next.) We will present more data on this issue in a later section.

## 4.2 Overall difficulty

As predicted, subjects were less successful at discovering Selector rules than Counter rules. Across the two "Given" conditions, only 1 of the 18 subjects with "Actual" Counter rules failed to discover the rule, while 5 of 18 failed to discover Selector rules. The proportion of successful subjects in each condition was: Counter #→# Counter - 100%, Selector #→# Counter - 92%, Counter #→# Selector - 67%, Selector #→# Selector - 83%. Thus, discovering a counter rule was easiest, whereas discovering a selector rule was more difficult. Also, switching from the selector frame to the counter frame was much easier than switching from the counter frame to the selector frame. Given that we already knew that subjects regard counter hypotheses as more likely than selectors (Klahr & Dunbar, 1988), the results of this study suggest that it is the *apriori* strength of belief in hypotheses that will determine how difficult it is to switch frames.

### 4.2.1 Trial of correct hypothesis

Another aggregate measure of the relative difficulty of the four conditions is the trial on which subjects arrive at the correct rule (i.e. the point when the Active and Actual hypotheses become the same.) As shown in the protocol listed in Appendix A, subjects usually state the Active hypothesis just before they write an experiment to test it. Thus, we can compute the proportion of subjects who arrive at the correct hypothesis prior to each experiment. Figure 4 shows the cumulative proportion of subjects in each condition who stated the correct hypotheses prior to the Nth experiment in their series. The effects of condition are very clear. Although a few subjects immediately reject the Given hypothesis and luckily guess the Actual rule prior to the first experiment, there is no reliable effect for such correct anticipations. By the second experiment, half of the subjects in both Actual = Counter groups have proposed the Actual rule, but none of the subjects in the Counter #→# Selector group have.

---

Insert Figure 4 about here

---

Figure 5 shows the proportion of subjects in each condition who stated the Actual hypothesis prior to the third experiment. All Counter #→# Counter subjects could make the minor revision in the preferred hypothesis necessary to go from Rule A to B or B to A by their third experiment. However, when subjects had to change from a Counter to a Selector only 42% of them were able to abandon a preferred counter for a selector by experiment 3. As noted above, the difficulty of frame change is asymmetric, as all but 1/4 of the Selector #→# Counter subjects have discovered that the unpreferred selector was wrong and have discovered the correct counter. Finally, even though no frame change is required, subjects have difficulty making the minor within-frame revision necessary in the Selector #→# Selector condition and 33% of them fail to do so by the third experiment. As Figure 4 shows, this relative order of difficulty remains for experiment 3 and beyond: Counter #→# Counter is relatively easy, Counter #→# Selector is relatively difficult, and the two Given=Selector conditions are roughly equivalent and of intermediate difficulty.

---

[7] In fact, all four rules *are* distinguishable under this program. Rules A, B, C, and D would FIRE 27, 21, 17, and 20 times, respectively. However, this is extremely difficult for the subject to figure out under these circumstances.

---

Insert Figure 5 about here

---

#### 4.2.2 Number of experiments run.

The success rate measures indicate that the frame change required by the Counter #→# Selector condition was particularly difficult. Another sensitive measure of difficulty is the number of experiments run. Recall that once subjects completed the three mandatory experiments, they were free to run additional experiments until they were satisfied that they had discovered the correct rule for RPT. As shown in Figure 6, only one-third of the Counter #→# Counter subjects chose to run a fourth experiment, and none ran more than 4. Half of the subjects in both the Counter #→# Selector and Selector #→# Selector conditions, and two-thirds of the Selector #→# Counter subjects ran 4 or more experiments. More of the subjects in between-frame conditions ran extra experiments than did subjects in within-frame conditions. The mean number of extra experiments per subject was 0.5 for the within-frame conditions and 1.3 for the between-frame conditions.

---

Insert Figure 6 about here

---

#### 4.2.3 Identical experiments.

When subjects are particularly surprised or confused by an experimental outcome, they occasionally repeat an experiment, i.e, write a program with the same λ-N combination as an earlier (usually immediately preceding) program. Although this is a relatively rare event, it provides another sensitive index of the relative difficulty of our experimental conditions. Out of the 150 experiments run overall, we observed 14 such pairs of identical experiments, and they occurred *only* in the frame-change conditions. For both Selector #→# Counter, and Counter #→# Selector there were 7 pairs. In most of these cases, the problem was that subjects misencoded the outcome of the first experiment, not because it was particularly complex, but because their expectations at some crucial point left them unprepared to notice an essential piece of behavior of the device.

### 4.3 Search in the Experiment Space

Although the legal range of values for both λ and N is from 1 to 15, subjects tended to be conservative in both the length of program they ran and the value of N. Over 90% of the experiments were within the $\lambda \leq 6$ by $N \leq 5$ E-space depicted in Figure 2, and more than 60% were within a 4 by 3 subset of that range. Each experiment was classified according to its location in the λ - N space shown in Figure 2. If subjects were selecting values of λ and N at random, then the expected relative frequency of experiments in each of the E-space regions would be proportional to the size of that region in the 6x5 E-space (Region 1: 6/30, Region 2: 10/30, Region 3: 14/30), and would be the same for all conditions. On the other hand, if subjects are sensitive to the interaction between the potential informativeness of different regions of the E-space and the hypothesis being tested, then we would expect to see an effect of frame-type and E-space region. More specifically, when the goal of hypothesis testing is to demonstrate an effect, subjects should design experiments that will highlight that feature. For Counter hypotheses, this focus would lead to an attempt to demonstrate that N controls the number of repetitions, which is best demonstrated by larger values of N. For small values of λ, this tends to produce programs in region 3. On the other hand, the clearest way to demonstrate a Selector hypothesis is to use a value of N that disambiguates the selected segment or step from first, last or all steps in a program. Region 2 is the preferred region for such demonstrations.

### 4.3.1 E-space distributions

Table 5 shows the distribution of experiments under three aggregations. The first two rows show the distribution on Experiment 1 as a function of the frame of the Given Hypothesis. On the first experiment, Region 2 is under-represented for Counter hypotheses and over-represented for Selector hypotheses, and the reverse is true for Region 3. The two distributions are significantly different from each other ($p < .005$) and from a random model. These results suggest that even on their very first experiments, subjects are sensitive both to the properties of the E-space and to the plausibility (to the subjects) of the Given hypothesis. When given a plausible Counter, subjects are very likely to construct a program with a large value of N, so as to get a clear effect of number of repetitions. On the other hand, when given an implausible Selector, subjects are more likely to start with an experiment that can clearly discriminate between Counters and Selectors.

**Table 5:** Distribution of Experiments in Experiment Space.

| | Given | 1 | 2 | 3 | N |
|---|---|---|---|---|---|
| First Experiment | Counter | 6 | 1 | 11 | 18 |
| | Selector | 6 | 10 | 2 | 18 |
| Third Experiment | Actual | | | | |
| | Counter | 6 | 7 | 4 | 17 |
| | Selector | 3 | 6 | 9 | 18 |
| Third Experiment | Frame Change | | | | |
| | No | 4 | 4 | 3 | 11 |
| | Yes | 5 | 9 | 10 | 24 |

Recall that for half of the subjects the Given hypothesis was from a different frame than the Actual hypothesis. Consequently, classification of experiments by frame of Given hypothesis becomes increasingly invalid as subjects start to discover that the Given hypothesis is incorrect and therefore design experiments to test an Active hypothesis from the Actual frame. Thus, the Given frame type could only be expected to have an effect for the first experiment. Indeed, by the second experiment, this pattern shown when classifying according to Given hypothesis disappears. Although the pattern reemerges for the third experiment, it is marginally significant [$p < .08$].

For the third experiment, programs were classified in two ways: by the Actual hypotheses (3rd and 4th rows in Table 5, and by whether a frame-change was necessary (5th and 6th rows). By the third experiment, neither the actual X region nor the frame-change X region distributions are significantly different from one another or from a random distribution of experiments in E-space *regions*. However, a finer-grained cell-by-cell analysis reveals a strong effect of frame-change. Eleven of the 24 frame-change subjects, but none of the 11 same-frame subjects[8] had third experiments in cells 3,2 or 3,3. These cells tend to be selected as a consequence of (a) the high discriminability of 3,2 and (b) the incremental VOTAT strategy described in the next section. By Experiment 3, subjects in the same-frame condition are more "daring" in their experiments: 5 of the 11 same-frame experiments have $\lambda > 4$, while only 3 of the 24 frame-change experiments are that long.

One possible reason why neither Actual Hypothesis nor Frame-Change conditions had any reliable effect on the region of Experiment 3 is that different subjects were at different points in their approach to the correct hypothesis by the third experiment. We analyzed the data according (a) whether subjects *ever* went into Region 2, and (b) what region subjects were in just prior to

[8]One subject wrote only 2 experiments.

their announcement of the correct hypothesis. There were two kinds of very clear regional effects. First, of the 30 subjects who were successful, 28 went into Region 2 at least once (.93), while 4 the 6 subjects who failed to reach the correct hypothesis never went into Region 2 (.67). The two who did go into Region 2 wrote programs that did not discriminate between the actual hypothesis and an idiosyncratic hypothesis that they held. Second, with respect to the region preceding the correct hypothesis, Actual=Counter subjects were in Region 2 55% of the time, while Actual=Selector subjects were there 71% of the time. Only 4 of the 30 successful subjects were in Region 3 immediately prior to announcing the correct hypotheses. Of these 4, 3 were in Actual=Counter groups where an experiment in Region 3 would be sensitive to variations in N, and therefore highly informative.

### 4.3.2 Incremental search in the Experiment Space

The analysis of Experiment Space regions gives a picture of the properties of experiments in isolation, but it does not reflect the nature of the incremental paths followed by subjects as they move from one experiment to the next. The VOTAT strategy mentioned earlier would lead to conservative moves in the E-space: moves that do not vary both $\lambda$ and N at the same time (including moves that vary neither). Overall, about half of the E-space moves were conservative, but they were more conservative in the frame-change conditions. Table 6 shows the proportion of conservative moves for each condition. The first column shows the proportion only for the first transition (i.e., between the first and second experiments) and the second column shows the proportion of all transitions that were conservative. It is clear that for the Counter #→# Counter condition, when both the Actual and the Given hypotheses are from the preferred frame, subjects are relatively bold in proposing their second experiment, and two-thirds of them changed both $\lambda$ and N. However in frame-change conditions, where the outcome of subjects' first experiment was highly discrepant with their expectations based on the Given hypothesis, subjects were much more conservative in moving about the Experiment Space: only 1/3 of them changed $\lambda$ and N simultaneously.

Table 6: Proportion of conservative transitions in E-space.

| Condition | First Transition | All Transitions |
|---|---|---|
| Counter-Counter | .33 | .43 |
| Selector-Selector | .50 | .44 |
| Counter-Selector | .67 | .64 |
| Selector-Counter | .67 | .64 |
| Mean | .54 | .53 |

### 4.3.3 Discriminating power of Experiments

In Section 2.3 we presented a formal analysis of the discriminating power of the different regions of the Experiment Space. In this section, we summarize the actual discriminating power of subject's experiments. Each experimental outcome was coded in terms of how many hypotheses were consistent with it. For each subject on each experiment we considered only the four hypotheses used in this study plus any idiosyncratic hypotheses that the subject may have mentioned. Then we computed, for each condition, the mean number of hypotheses that would be consistent with each experimental outcome (averaged over all the subjects in the condition.) The results are listed in Table 7. While three of the groups are able to write programs that eliminate all but one hypotheses, the Counter #→# Selector subjects design experiments at the outset whose outcomes are consistent with between two and three hypotheses, and even by their third experiment, they are just approaching the First Experiment mean of the other three groups.

Another way of describing the discriminating power of subjects' search of the E-space is in terms of the regions that are *avoided* while testing particular hypotheses. For First experiments all subjects avoided an N=1, $\lambda$=1 experiment, as it would not discriminate among any of the

**Table 7:** Mean number of hypotheses consistent with experimental outcomes.

| | Experiment Number | | | |
|---|---|---|---|---|
| Condition | 1 | 2 | 3 | 4 |
| Counter-Counter | 1.5 | 1.0 | 1.2 | 1.0 |
| Selector-Selector | 1.7 | 1.3 | 1.2 | 1.0 |
| Counter-Selector | 2.8 | 1.8 | 1.6 | 1.5 |
| Selector-Counter | 1.6 | 1.3 | 1.1 | 1.1 |

hypotheses. Two-thirds of the Given = Counter subjects conducted experiments that could distinguish between the other counter hypothesis, suggesting that they were testing more than one hypothesis at a time and were avoiding undiscriminating regions of the experiment space. All of the Given = Selector subjects conducted first experiments in regions that would discriminate between one selector hypothesis and another.

These results suggest that when given an hypothesis to test, subjects do consider other hypotheses within that frame, and write programs that would allow them to discriminate between same-frame alternatives. If subjects are only considering hypotheses within the frame of the given hypothesis, then we should expect to see many experiments that would not distinguish between hypotheses from different frames. In fact 47% of first experiments cannot rule out specific hypotheses from the alternate frame. If we break this down further, we find that when given Counters only 33% of first programs can rule out (or confirm) selectors, whereas when given Selectors 66% of programs could rule out (or confirm) counters. Again, this reflects that the *apriori* belief that BT works like a counter is an important factor in determining what parts of the experiment space to search.

## 5 Discussion

Overall, these results suggest that subjects are remarkably adept at designing and interpreting experiments in a novel domain. When subjects are given a plausible hypothesis, they tend to design an experiment that demonstrates the effect that is to be expected. When given implausible hypotheses, they write programs that are good discriminators. When the discrepancy between the Given and the Actual hypothesis is very great, subjects are more conservative in moving from one experiment to the next. The fundamental question for builders of computational models of the experimental design process is how subjects bring to bear general heuristics for "good experiments" on this novel domain.

### 5.1 Hypothesis Generation Heuristics

Any scientific enterprise is conducted in the context of the currently available knowledge of the domain: initial hypotheses are determined by the knowledge of the domain. In the case of the BT domain, almost all of the commands that are learned in the initial phase work by executing a command N times. As a result, subjects are initially predisposed toward hypotheses that are counters. This is evident in the results of this study and our previous work (Klahr & Dunbar, 1988). The study discussed in this chapter also suggests that subjects consider more than one hypothesis at a time: Both the subject protocols and the types of experiments conducted suggest that the subjects consider various hypotheses within a frame. Thus, one heuristic used is that of generating a frame and then generating various slot values within that frame. Then experiments are conducted that will discriminate between rival hypotheses within the frame. The data also suggest that it is easy to think of hypotheses from an alternate frame, but only when the strength of belief in the current frame is less than that of the alternate frame. Thus subjects in our Selector #→# Counter group were much more successful than in the Counter #→# Selector group.

These findings suggest that a useful heuristic in a computational model of experiment generation would be to initially generate different frames and conduct experiments that distinguish between frames, rather than designing experiments that discriminate between rival hypotheses that are all from the same frame. This heuristic is slightly different from the one that is usually used in discussions of scientific methodology. The usual claim is that multiple hypotheses should be considered when designing an experiment, here we are arguing that this is most effective when the alternate hypotheses come from different frames. Once the frame is established then the correct slot values of the frame can be determined. Essentially, we are advocating a form of breadth first search.

Psychologically, the fact that subjects in this study were testing multiple hypotheses, whereas subjects in our previous work and the work of others (e.g., Mynatt et al. 1977), were not testing multiple hypotheses is revealing. In this study, subjects were given hypotheses to test, whereas in most other studies subjects must generate their own initial hypotheses. This difference in procedure had two effects. First, subjects almost always generated hypotheses other than the one given, resulting in the testing of multiple hypotheses. Second, subjects abandoned the given hypothesis much more readily than if they had generated the hypotheses themselves. In the Klahr and Dunbar (1988) study most subjects initial (self-generated) hypothesis was A. They only discovered that BT worked according D after 15 experiments. In this study, two of the three subjects in the A #→# D group discovered that it worked according to D after only 4 experiments. These results suggest that self-generated hypotheses are given higher strength values than externally generated hypotheses -- a fact that becomes apparant when articles are submitted for publication!

## 5.2 Experiment generation heuristics

While the BT domain may appear relatively simple in comparison to that faced by a scientist in a laboratory, the size of the BT experiment space is surprisingly large. As we noted in our earlier work (Klahr & Dunbar, 1988) there are nearly 500 billion distinct experiments that subjects could potentially conduct. Even if we limit the space of experiments to programs that have a length of 4 instructions (using 4 of the 8 distinct commands) there is a space of around 1700 experiments that could be designed. Most subjects appear to understand immediately that specific instructions are not important, and that only the N - $\lambda$ space is relevant, but even it can be as large as 225 cells (15 x 15). Thus, when asked to write only three experiments, subjects must prune this spac effectively. There is clear evidence that subjects do manage to drastically prune the space. As noted earlier, 60% of the experiments occur within a $\lambda \leq 4$, $N \leq 3$ area of the E-space, although it represents only 5% of the 15 x 15 E-space. Even within this preferred area, experiments are not uniformly distributed. The 1,1 cell is never used, presumably because subjects realize that it provides no information. Conversely, the 3,2 cell is disproportionately selected 18 times out of 106 total programs in the first three experiments. This is 5 times more than expected in a random selection from a 6 x 5 space and twice the expected frequency in a 4 x 3 space. This cell represents the minimum values of $\lambda$ and N in the maximally-informative region 2.

What enables subjects to be so effective in constraining their search in the E-space? We believe that the following heuristics are operating:

1. Maintain observability. Given that BT moves along the screen from one location to another, there is no permanent record of behavior and subjects must remember what BT actually did. Thus, one heuristic is to write short programs, making it possible to remember what happened and compare the results to those predicted by the Active hypotheses. Other uses of this heuristic are: use small values of N on going forwards or backwards (this is easy to see, and BT does not go off the screen); make turns that are easy to see, such as right-angles, and 180 degree turns.

2. Design experiments giving "characteristic" results. In the BT domain, this translates into "use distinct commands." Almost all subjects attempted to write programs

where every command was different. This makes it possible to determine what specific commands were repeated and the order that they were repeated in. This heuristic substantially reduces the size of the experiment space, while at the same time maximizes the observability of the programs. As Figure 3, and the protocol from subject MA quoted earlier shows, when all the commands are the same, it is extremely difficult to discriminate between rival hypotheses.

3. Focus on one dimension of an hypothesis. Most hypotheses are complex entities and have many aspects that can be focussed upon. Auxiliary hypotheses, anciliary hypotheses, and additional assumptions that are not tested must be made (cf. Lakatos & Musgrave, 1970) . That is, in going from an hypothesis to an experiment what is thought to be crucial will be focussed upon. Our results show that in the BT domain subjects tend to focus on one dimension of an hypothesis at a time. For example, when given a Counter hypothesis, subjects initially focus on the number of times something is repeated rather than what is repeated. This heuristic means that subjects miss some of the features of an experimental result, as they are only considering the result in terms of the current dimension of the hypothesis that is being focussed upon. Furthermore, the finding that many experiments change only one feature of the experiment at a time suggests that the focus is not only on one aspect of an hypothesis, but also on one aspect of an experiment. As we mentioned previously, this is a strategy that has been often discussed in the concept attainment literature (e.g., Tschrigi, 1980; Bruner et. al. 1956).

4. Exploit surprising results. Another experimental heuristic to use is to set up a new goal when a surprising finding occurs: When an unexpected result occurs set a new goal of tracking down the source of the unexpected finding. The subjects in the Counter #→# Selector condition who succeeded used this heuristic. They focussed on why the program was not repeated N times and changed their goal from trying to fit the result into a counter frame, to using the surprising experimental result to induce new hypotheses. Subjects in this condition that did not use this strategy continued to focus on how many times things were repeated, rather than focus on the surprising result. Dunbar (1989), in a study that simulated a discovery in a genetics experiment, also found that only subjects who used the strategy of generating a new goal of explaining surprising results were able to discover the mechanism underlying genetic control.

   Kulkarni and Simon (1988) noted that this heuristic was used by Krebbs in his discovery of the Ornithine cycle. They have instantiated this heuristic in their program KEKADA. Holland, Holyoak, Nisbett, and Thagard (1986) have also suggested that the generation of new hypotheses from surprising findings (abduction) is a useful inductive procedure and they have instantiated it in their PI program. However our results suggest that the performance of subjects who follow up surprise results is more similar to the setting up of a new problem space when an obstruction is encountered. This is the learning mechanism underlying SOAR (Newell, 1988). Using this heuristic, a surprising result is not used to immediately propose a new hypothesis, rather the surprising result is used to set up a new problem space. The focus of experimentation in the new problem space may be different (e.g., in the BT context; shifting focus from how many times something is repeated, to what is repeated). This shift of focus will result in a shift to a new region of the experiment space and eventually to the generation of new hypotheses. We have also discussed a group of subjects that use this heuristic (Experimenters) in Klahr and Dunbar (1988).

5. Use apriori strength of an hypothesis to choose experimental strategy. One of the most often discussed issues in the literature on scientific reasoning has been that subjects tend to try and confirm, rather than disconfirm their current hypothesis (cf. Klayman & Ha, 1987). The study discussed in this chapter reveals that the

strategies of confirmation and disconfirmation vary with the strength of the belief on the currently held hypothesis. When the hypothesis is thought to be highly likely, subjects often set themselves the goal of demonstrating the key features of the given hypothesis, rather than conducting experiments that can discriminate between a large number of hypotheses. A less common strategy for highly likely hypotheses was to use the RPT key as a subgoal to perform an action, for example drawing a square. In another study (Dunbar, Klahr, & Fay, 1989) we found that young children frequently use this strategy. For hypotheses with low apriori strength subjects use a different strategy. In this case subjects usually propose hypotheses from frames other than the given frame, and conduct experiments that will discriminate between rival hypotheses. Subjects search the Hypothesis space before conducting any experiments and when they design an experiment, they select an experiment that is in a region of the experiment space that can potentially disconfirm the hypothesis that they are testing.

Not only do subjects appear to use these heuristics, but also they appear to be able to deal with their inherent contradictions. As we noted earlier, no subject ever uses the 1,1 cell, even though it would yield the easiest to observe behavior, because it is so uninformative with respect to discriminating among rival hypotheses. On the other hand, the frequent use of the 3,2 cell represents a minimax solution to the conflicting heuristics of minimizing cognitive load and maximizing discriminability. We are not suggesting that subjects are able to carry out an optimization algorithm that selects this solution. Instead, we believe that the interaction of multiple heuristics produces, in our subjects, the same kind of behavior that Giere (1988) describes in terms of "the scientist as satisficer".

## 6 Conclusion

As we stated at the beginning of this chapter, our work starts not with the construction of computational systems for conducting scientific discovery, but with an analysis of the performance of subjects as they generate experiments to test hypotheses. The results of the study discussed in this chapter suggest a number of powerful heuristics that can be used to design experiments and formulate new hypotheses. Some of these heuristics are very successful and lead toward discovery. For example, generating hypotheses from alternative frames, and setting new goals of explaining surprising results led toward the discovery of the correct hypothesis and resulted in fewer experiments. Other heuristics that subjects used tended to be less effective; searching for confirmation, focussing on hypotheses within one frame.

Some of the 'good' heuristics that we have discovered are similar to those that have been discovered in the other approaches to scientific reasoning that we mentioned earlier -- historical analyses of scientific discovery (Kulkarni and Simon, 1988; Darden, 1987), and computational models (Langley, Simon, Bradshaw, and Zytkow, 1987; Holland, Holyoak, Nisbett, and Thagard, 1986). This is encouraging, as it suggests that we are coming closer to an understanding of the processes underlying scientific discovery. However, as Klayman and Ha (1987) have noted, certain hypothesis testing methods that are useful in one context may be totally inappropriate in other contexts. Thus, a further goal for our research is to discover the contexts under which heuristics should, and should not be used.

# Figure Captions

**Figure 1**: Frames for hypotheses about how RPT N works. Heavy borders correspond to common hypotheses from Table 1; dashed borders correspond to partially specified hypotheses; arrows indicate that adjacent hypotheses differ along a single attribute shown on the arrow; all possible hypotheses are not shown.

**Figure 2**: Regions of the Experiment Space, showing illustrative programs and confirmation/disconfirmation for each common hypothesis. (Shown here is only the 6x5 subspace of the full 15x15 space.)

**Figure 3**: Behavior of BigTrak under four different rules and programs from each of the Experiment Space Regions. Each row shows a generic program and how BigTrak would behave under each of the four Rules used in this study. For each entry, executions under control of RPT are shown in boldface. The upper portion of the Figure shows programs using distinct commands; the lower portion shows programs using identical commands. [See text for further explanation. ]

**Figure 4**: Proportion of subjects generating correct hypothesis by Nth experiment

**Figure 5**: Proportion of subjects generating correct hypothesis by Third experiment

**Figure 6**: Proportion of subjects running N experiments.

## References

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

Darden, L. (1987). Viewing the history of science as compiled hindsight. *Artificial Intelligence, 8*(2), 33-41.

Dunbar, K. (1989). Scientific reasoning strategies in a simulated molecular genetics environment. In *Proceedings of the Eleventh Annual Meeting of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum. In press.

Dunbar, K., & Klahr, D. (1989). Developmental differences in scientific discovery strategies. In D. Klahr & K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon*. Hillsdale, NJ: Erlbaum.

Dunbar, K., Klahr, D., & Fay, A.L. (1989). Developmental differences in scientific reasoning processes. Paper presented at the Biennial Meetings of the Society for Research in Child Development, Kansas City, MO, April 27-30.

Giere, R.N. (1988). *Explaining science: a cognitive approach.* Chicago, IL: University of Chicago Press.

Holland, J., Holyoak, K., Nisbett, R.E., & Thagard, P. (1986). *Induction: Processes of Inference, Learning, and Discovery.* Cambridge, MA: MIT Press.

Klahr, D., & Dunbar, K. (1988). Dual space search during scientific reasoning. *Cognitive Science, 12*(1), 1-55.

Klayman, J., & Ha, Y. (1985). Hypothesis testing in rule discovery: Strategy and structure. Paper presented at the Tenth Research Conference on Subjective Probability, Utility, and Decision Making, Helsinki, Finland, August 1985.

Klayman, J., & Ha, Y. (1987). Confirmation, disconfirmation and information in hypothesis testing. *Psychological Review, 94*, 211-228.

Kulkarni, D., & Simon, H.A. (1988). The processes of scientific discovery: The strategy of experimentation. *Cognitive Science, 12*, 139-175.

Lakatos, I., & Musgrave, A. [Eds.]. (1970). *Criticism and the growth of knowledge.* New York, NY: Cambridge University Press.

Langley, P., Simon, H.A., Bradshaw, G.L., & Zytkow, J.M. (1987). *Scientific discovery: Computational explorations of the creative processes.* Cambridge, MA: MIT Press.

Minsky, M. (1975). A framework for representing knowledge. In P.H. Winston (Ed.), *The psychology of computer vision* (pp. 211-277). New York: McGraw-Hill.

Mynatt, C. R., Doherty, M. E., & Tweney, R.D. (1977). Confirmation bias in a simulated research environment: an experimental study of scientific inference. *Quarterly Journal of Experimental Psychology, 29*, 85-95.

Newell, A. (1988). The 1987 William James Lectures: Unified theories of cognition. Departments of Computer Science and Psychology, Carnegie-Mellon University.

Newell, A. (1989). Putting it all together: Final comments. In D. Klahr & K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon*. Hillsdale, NJ: Erlbaum.

Qin, Y., & Simon, H.A. (1989). Laboratory replication of scientific discovery processes. *Cognitive Science.* In press.

Shrager, J. (1985). *Instructionless learning: Discovery of the mental model of a complex device.* Doctoral dissertation, Department of Psychology, Carnegie-Mellon University.

Shrager, J., & Klahr, D. (1986). Instructionless learning about a complex device. *International Journal of Man-Machine Studies, 25*, 153-189.

Simon, H.A., & Lea, G. (1974). Problem solving and rule induction: A unified view. In L.W. Gregg (Ed.), *Knowledge and cognition* (pp. 105-128). Hillsdale, NJ: Erlbaum.

Tschirgi, J.E. (1980). Sensible reasoning: A hypothesis about hypotheses. *Child Development, 51*, 1-10.

# Appendix A

Example of a complete protocol. (DP26) CLR and GO commands have been deleted. BigTrak's Behavio, is shown in boldface type.

Given: A - Repeat entire program N times.
Actual: C - Repeat Nth step once.

002 **What I want to do is um...**

003 **I want to test to see if repeat repeats the statement before it,**

004 **so I'll write just a little program that does something,**

005 **say turns left ten, and then put the repeat statement in**

006 **and we'll just repeat it twice 'cause it's not really important**

007 **for the first program.**

008 **And then just to check I'll put in a statement after the repeat**

009 **to see if it has any effect on that,**

010 **so it's just left 10, repeat it twice and then right ten.**

011 **EXP: OK, SO WHAT ARE THE COMMANDS?**

012 **Left ten, repeat two, and right ten.**

013 **Experiment 1:** ← 10 **RPT 2** → 10

← 20 → 10

014 **It did what you would expect it would do,**

015 **it did repeat the left ten twice so it turned left twenty**

016 **and then back to the right ten,**

017 **so it appears that the repeat doesn't have any effect on any statements**

018 **that come after it,**

019 **so I guess now we'll try putting more than 1 statement**

020 **in front of the repeat and see what happens**

021 **we'll do left ten and forward two, repeat twice,**

022 **and just to check it again at the end we'll have it fire once**

023 **and see if it repeats or not**

024 **It's left ten, forward two, repeat twice, fire one.**

025 **Experiment 2:** ← 10 ↑ 2 **RPT 2 FIRE 1**

← 10 ↑ 4 FIRE 1

026 **Ok, it only turned left once,**

027 but it went forward four,
028 so it looks like it only repeats the statement immediately in front
029 of it.
030 I guess for the last test we'll just try a more complicated program
031 with the repeat in the middle of it
032 and see what statements are repeated
034 So this is just a long program with a repeat in the middle of it
035 to see what statements are repeated.
036 EXP: OK, SO WHAT ARE THE COMMANDS?
037 Forward one, right fifteen, back one, repeat two, left ten,
038 forward 1.
039 Experiment 3: ↑ 1 → 15 ↓ 1 RPT 2 ← 10 ↑ 1
040

↑1 →15 ↓1 →15 ←10 ↑1

046 Ok, that time it repeated the uh.. the right fifteen command
047 EXP: SO DO YOU KNOW HOW IT WORKS,
048 OR WOULD YOU LIKE TO WRITE MORE PROGRAMS?
049 I'm not really sure how it works,
050 because it seemed to act differently in number two, and number 3.
051 It would be best to write more.
052 Should I just go ahead?
053 EXP: YES JUST WRITE NUMBER FOUR.
054 Let's just try the same program as before,
055 except we'll put a three in, repeat three instead of repeat 2,
056 and see if that has anything to do with it.
057 Forward one, right fifteen, back one, repeat three, left ten,
058 forward one
059 Experiment 4: ↑ 1 → 15 ↓ 1 RPT 3 ← 10 ↑ 1
060

↑1 →15 ↓2 ←10 ↑1

061 That time it repeated the statement here but it repeated it twice,
062 the number four.. the number three..

063 it repeated the turning statement once.

064 EXP: WOULD YOU LIKE TO TRY ANOTHER PROGRAM?

065 I guess, I don't really have any idea of what it's doing

066 because when I change the number not only did it change..

067 it didn't change the uh..

068 the number that it repeated

069 but it changed the uh.. the actual instruction.

070 I guess my conjecture is, right now, that it says repeat two

071 so it repeats the second instruction,

072 and here it repeats three and it repeats the third instruction.

073 So we'll try the same thing with repeat one,

074 and see if it repeats the first instruction.

075 Forward one, right fifteen, back one, repeat one, left 10, forward one

076 Experiment 5: ↑ 1 → 15 ↓ 1 RPT 1 ← 10 ↑ 1

077

↑1 →15 ↓1 ↑1 ←10 ↑1

078 Ok, I think I know what it does now.

079 EXP: OK..

080 When it hits the repeat statement..

081 when it says repeat one it means at this point repeat statement

082 number one

083 and in this case it went forward and it turned and it went back

084 and then it came forward again, which is the first statement.

085 and it did something similar, I mean it went forward one, turned right

086 went back, and it hit repeat three and this is the third statement

087 so it went back to here

088 EXP: OK, SO HOW, IN GENERAL, DOES THE REPEAT KEY WORK?

089 If you type, it looks, when it hits the repeat statement,

090 if you look through the program when there's like repeat six

091 it takes the sixth statement and does that,

092 when it hits the repeat statement it'll repeat the sixth statement.

093 EXP: OK, GREAT.

# Appendix B

Example of a complete protocol. (JS02) CLR and GO commands have been deleted. BigTrak's Behavior is shown in boldface type.

Given: A - Repeat entire program N times.
Actual: C - Repeat Nth step once.

002 **Alright, Program 1, if that is the hypothesis,**

003 **which I'm not so sure, if it's not the hypothesis**

004 **I'm going to design a program**

005 **that's going to prove that it's not the hypothesis,**

006 **and I think a good way of doing that would be a simple program,**

007 **so, uh.. I'm going to put in first move forward one**

008 **and uh.. that's just a good way to start off**

009 **and I want distinct steps here to see if it is repeating it**

010 **so I will have a right turn, fifteen degrees**

011 **then I think a good maneuver here would be just to have it fire once**

012 **it's just something that's distinguished.**

013 **Then to see if this thing moves like this**

014 **it might go in a reverse order**

015 **or it might just repeat the step number**

016 **but I sort of doubt that**

017 **because there's no numbered lines to these programs**

018 **I'm going to have it move backwards one**

019 **and that will put it back to the left facing forward**

020 **and then we will try a repeat which will..**

021 **we'll try to repeat one**

022 **repeat one will bring it back to its original position**

023 **but it will be facing the opposite direction**

024 **so after the repeat one**

025 **to see what happens to the instructions that happen afterwards**

026 **we will put a turn left, thirty degrees**

027 **EXP: I'M GOING TO HOME IT AND CLEAR IT, NOW YOU TELL ME WHAT TO PRESS**

028 **Ok, up one, to the right fifteen, fire one, backwards one, repeat one**

030 **left thirty.**

031 **Experiment 1:** ↑ 1 → 15 FIRE 1 ↓ 1 RPT 1 ← 30

032 EXP: NOW WHAT DO YOU THINK MIGHT HAPPEN WHEN I PRESS GO
033 Um, well I think it's definately going to execute the first part of it
034 it's going to end up facing to the right
035 but over one block to the left of the position it's in now
036 and, uh, then if the hypothesis for the repeat is correct
037 then it will continue with the rest of the program
038 if it's not, the only thing I'm thinking it might do
039 is I think it might just move forward 1
040 and then it'll end up turning to the left 30, reversing it's direction
041 EXP:OK, I'M PRESSING GO

↑ 1 → 15 FIRE 1 ↓ 1 ↑ 1 ← 30

042 Aha, that's what I thought it would do
043 but that's not what the hypothesis said.
044 EXP: SO WHAT ARE YOU THINKING?
045 Well it's the original idea,
046 it's uh.. if I ran this same program and I said repeat two
047 it would repeat the second step.
048 if I said repeat three, it's going to fire again.
049 It's repeating the order of the steps that I put in,
050 I think.
051 Or, it might,
052 I want to try something here,
053 EXP: WHAT ARE YOU THINKING?
054 Well I'm thinking it might also be..
055 It'll repeat the first step..
056 If I put two, it might repeat the first and the second step
057 so I'm going to try two
058 actually I'll try three.. no I'll try two.

059 Do I have to write this whole program in again?

060 EXP: WHATEVER PROGRAM WE'RE GOING TO DO, YOU NEED TO WRITE IT FOR ME

061 I want to run the same program,

062 because I know what it does,

063 I just want to change the condition of the repeat

064 because I want to see if it's going to repeat

065 the first two instructions

066 or it's just going to repeat the second instruction

067 so we will give it a.. actually we'll give it a three

068 because if it's the first condition

069 then, um.. if it's my first idea

070 it's going to repeat just the third step

071 then I'll have to worry about it turning fifteen degrees

072 it just, it'll be easier

073 EXP: HOME CLEAR NOW WHAT?

074 Up one, to the right fifteen, fire one, backwards one, repeat three,

075 to the left thirty

076 Experiment 2: ↑ 1 → 15 FIRE 1 ↓ 1 RPT 3 ← 30

↑ 1 → 15 FIRE 1 ↓ 1 FIRE 1 ← 30

077 So it's just repeating the step number of the..

078 the number you put after the repeat it repeats that sequence,

079 that third (unintelligible) the fire or the turn right

080 or the turn left

081 it doesn't repeat first second and third like I thought it might,

082 it just repeats the third step.

083 EXP: CAN YOU WRITE ONE MORE PROGRAM TO BE SURE?

084 Yeah I'll write one more program.

085 Ok, this has some interesting things,

086 we'll make it move backwards three,

087 we will make it turn to the right sixty,

088 and we'll make it turn to the left..

089 no we want it to go.. we'll have it go straight ahead two

090 and we'll have it fire.. fire five times

091 and then we'll have it, let's see what I want to repeat here..

092 and then we'll have it do a nice little spin,

093 I'm curious, one two three four

094 fifth instruction, we'll make it..

095 (unintelligible) the sixth instruction,

096 I doubt it but I'm curious

097 because that would be the one that's after this,

098 One two three four five, um..

099 and the sixth instruction will be um..

100 what could we make it do interesting..

101 We don't have any backward.. yes we do have a backwards

102 something different, we will make it turn left ten

103 EXP: HOME CLEAR

104 Backwards three, right sixty, forward two, fire five, repeat six,

105 left ten

106 Experiment 3: ↓ 3 → 60 ↑ 2 FIRE 5 RPT 6 ← 10
↓3 → 60 ↑ 2 FIRE 10 ← 10

107 I wonder if it did that because repeat six,

108 since six didn't occur yet,

109 I should have put another step in there,

110 because six didn't occur yet

111 it might not actually be repeating the sixth one,

112 it may just be going on,

113 but that doesn't disprove anything anyway, it's just a thought

114 EXP: SO WHAT ARE YOU THINKING?
115 Well I think that whatever number you put after
116 it repeats that instruction line,
117 I set up this third program to prove that again,
118 but what I was curious about when I designed this program,
119 is whether it would repeat something it actually hadn't done yet.
120 Do you know what I'm saying?
121 Because so far it had moved backwards and turned around
122 and gone forward, it had fired five times,
123 then I'm asking it to repeat the sixth step in the program,
124 but the sixth step hadn't occurred.
125 Now what I should have done,
126 is I should have included another instruction
127 I should have had it repeat the seventh step
128 and put in a sixth instruction that was different,
129 because I don't know, from my last program,
130 I don't know whether,
131 I know that the number six means it'll repeat the sixth instruction
132 but, since it hadn't done it yet,
133 I don't know whether it went to the sixth one..
134 because of the repeat six,
135 or it said repeat six is an illegal quantity to put in there,
136 therefore we go on to the next instruction
137 and it just did the sixth instruction anyway
138 EXP: ARE YOU REALLY SURE YOU KNOW HOW IT WORKS,
139 EXP: OR DO YOU WANT TO WRITE ANY OTHER PROGRAMS TO BE SURE?
140 I'm really sure..
141 I'm curious about the last thing
142 whether it will actually repeat something that hasn't occured yet
143 EXP: BUT YOU'RE FAIRLY SURE YOU KNOW HOW IT WORKS?
144 yes
145 EXP: OK WHY DON'T WE STOP THERE THEN

# HYPOTHESIS SPACE

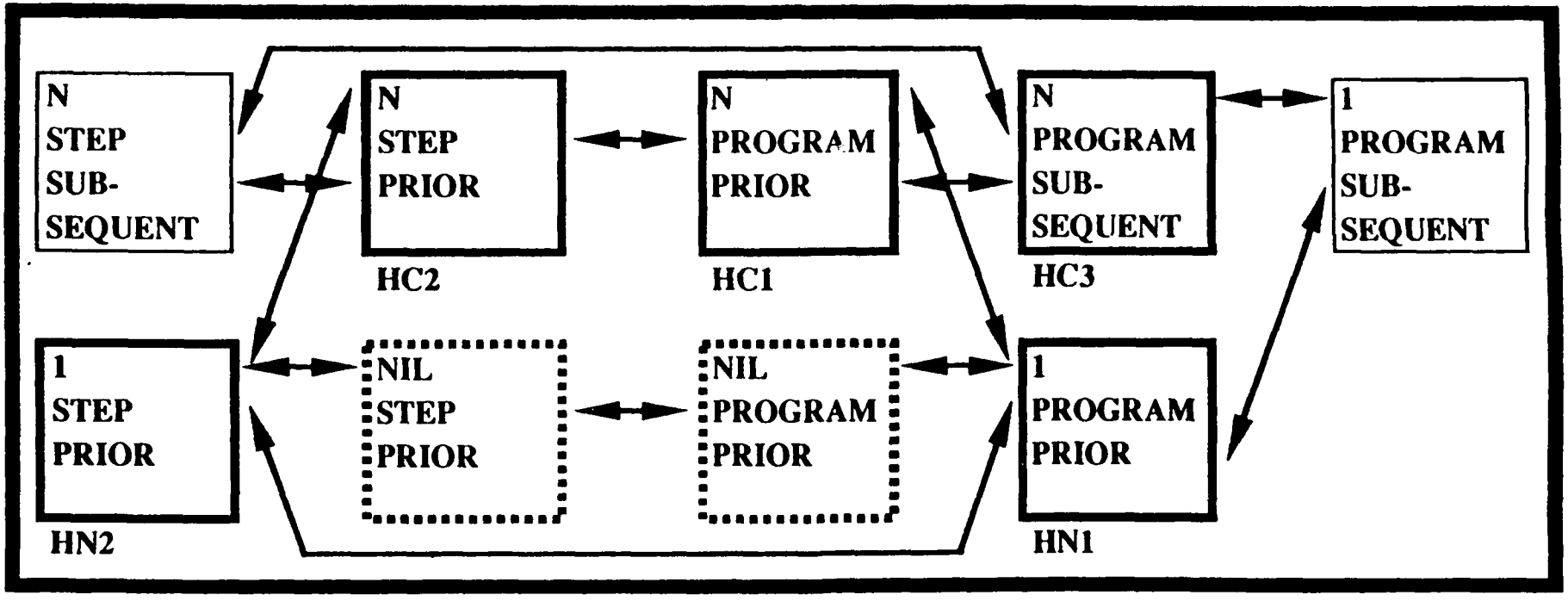
RPT

NROLE:
NREPS:
UNIT:
BOUNDS:

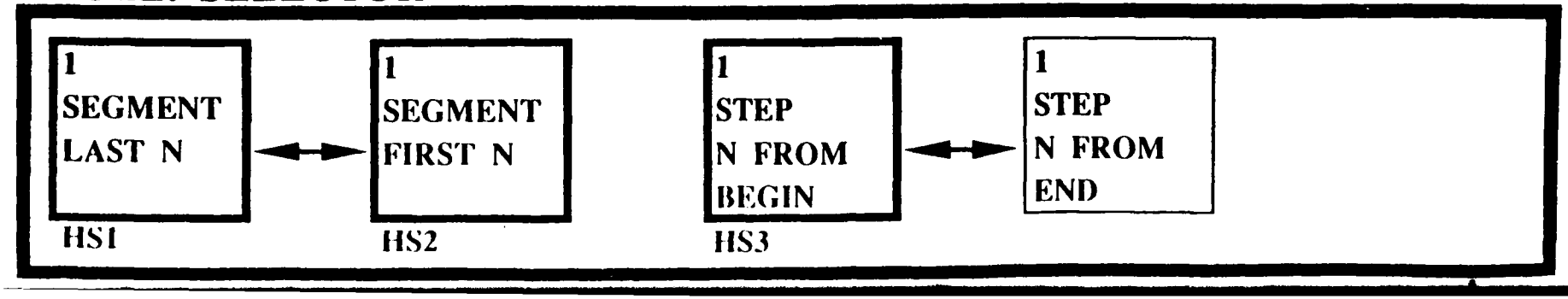
KEY

COMMON (WITH LABEL)
EXAMPLE OF IDIOSYNCRATIC
PARTIALLY SPECIFIED
CHANGED ATTRIBUTE

## NROLE: COUNTER




## NROLE: SELECTOR



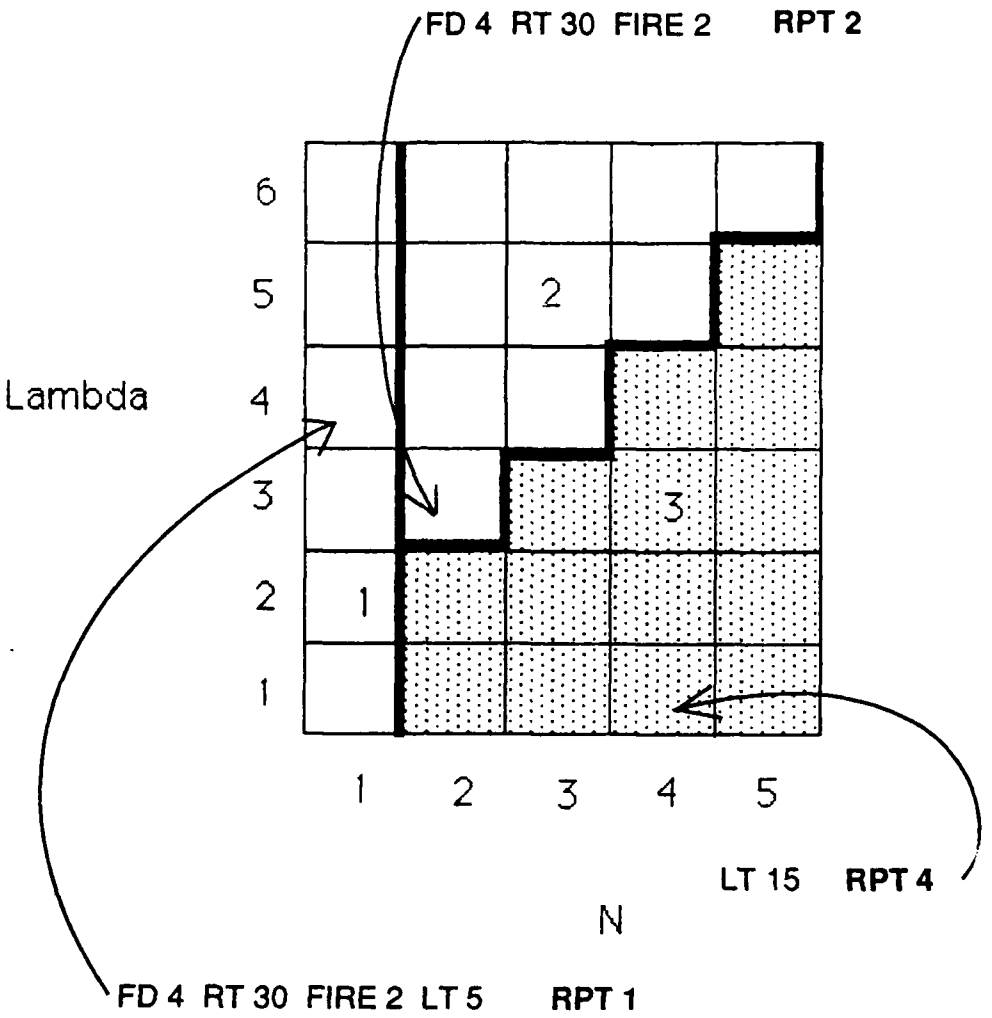

FD 4 RT 30 FIRE 2 RPT 2
Lambda
6
5
4
3
2
1
2
3
1
1 2 3 4 5
LT 15 RPT 4
N
FD 4 RT 30 FIRE 2 LT 5 RPT 1

Fig 2

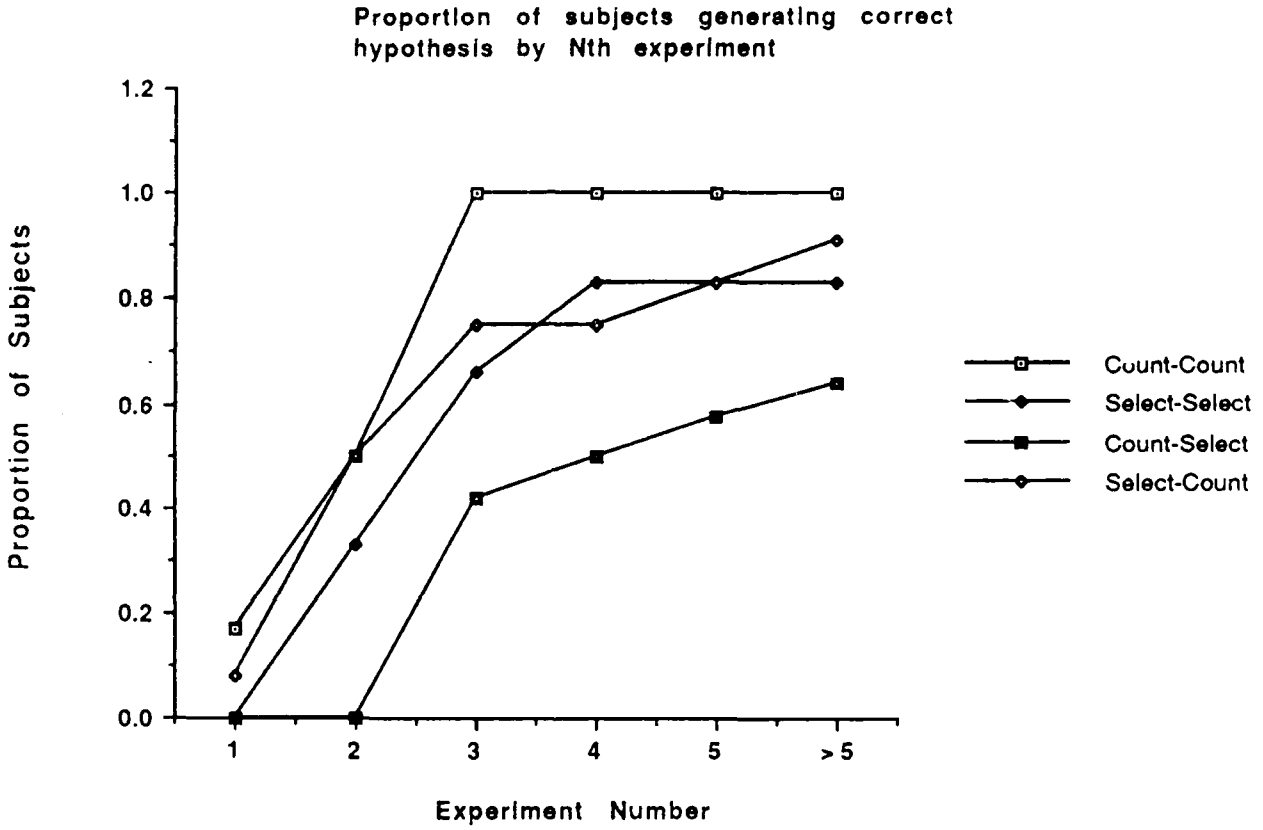
# BigTrak behavior under four different rules and different types of programs

*Highly Distictive Programs*

| | | | RULE | | | |
|---|---|---|---|---|---|---|
| | | | COUNTERS | | SELECTORS | |
| | | | A | B | C | D |
| No. | Region | Program | program N times | last step N times | Nth step once | last N steps once |
| 1 | 1 | X R1 | xx | xx | xx | xx |
| 2 | 1 | X Y R1 | xyxy | xyy | xyx | xyy |
| 3 | 1 | X Y Z R1 | xyzxyz | xyzz | xyzx | xyzz |
| 4 | 2 | X Y Z R2 | xyzxyzxyz | xyzzz | xyzy | xyzyz |
| 5 | 3 | X Y Z R3 | xyzxyzxyzxyz | xyzzzz | xyzz | xyzxyz |
| 6 | 3 | X Y R3 | xyxyxyxy | xyyyy | xyy* | xyxy* |
| 7 | 3 | X R3 | xxxx | xxxx | xx* | xx* |
| 8 | 3 | X Y Z R4 | xyzxyzxyzxyzxyz | xyzzzzz | xyzz* | xyzxyz* |

*Minimally Distinctive Programs*

| | | | A | B | C | D |
|---|---|---|---|---|---|---|
| No. | Region | Program | program N times | last step N times | Nth step once | last N steps once |
| 9 | 1 | X R1 | xx | xx | xx | xx |
| 10 | 1 | X X R1 | xxxx | xxx | xxx | xxx |
| 11 | 1 | X X X R1 | xxxxxx | xxxx | xxxx | xxxx |
| 12 | 2 | X X X R2 | xxxxxxxxx | xxxxx | xxxx | xxxxx |
| 13 | 3 | X X X R3 | xxxxxxxxxxxx | xxxxxx | xxxx | xxxxxx |
| 14 | 3 | X X R3 | xxxxxxxx | xxxxx | xxx* | xxxx* |
| 15 | 3 | X R3 | xxxx | xxxx | xx* | xx* |
| 16 | 3 | X X X R4 | xxxxxxxxxxxxxxx | xxxxxxx | xxxx* | xxxxxx* |


Proportion of subjects generating correct hypothesis by Nth experiment
Proportion of Subjects
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
>5
Experiment Number
Count-Count
Select-Select
Count-Select
Select-Count

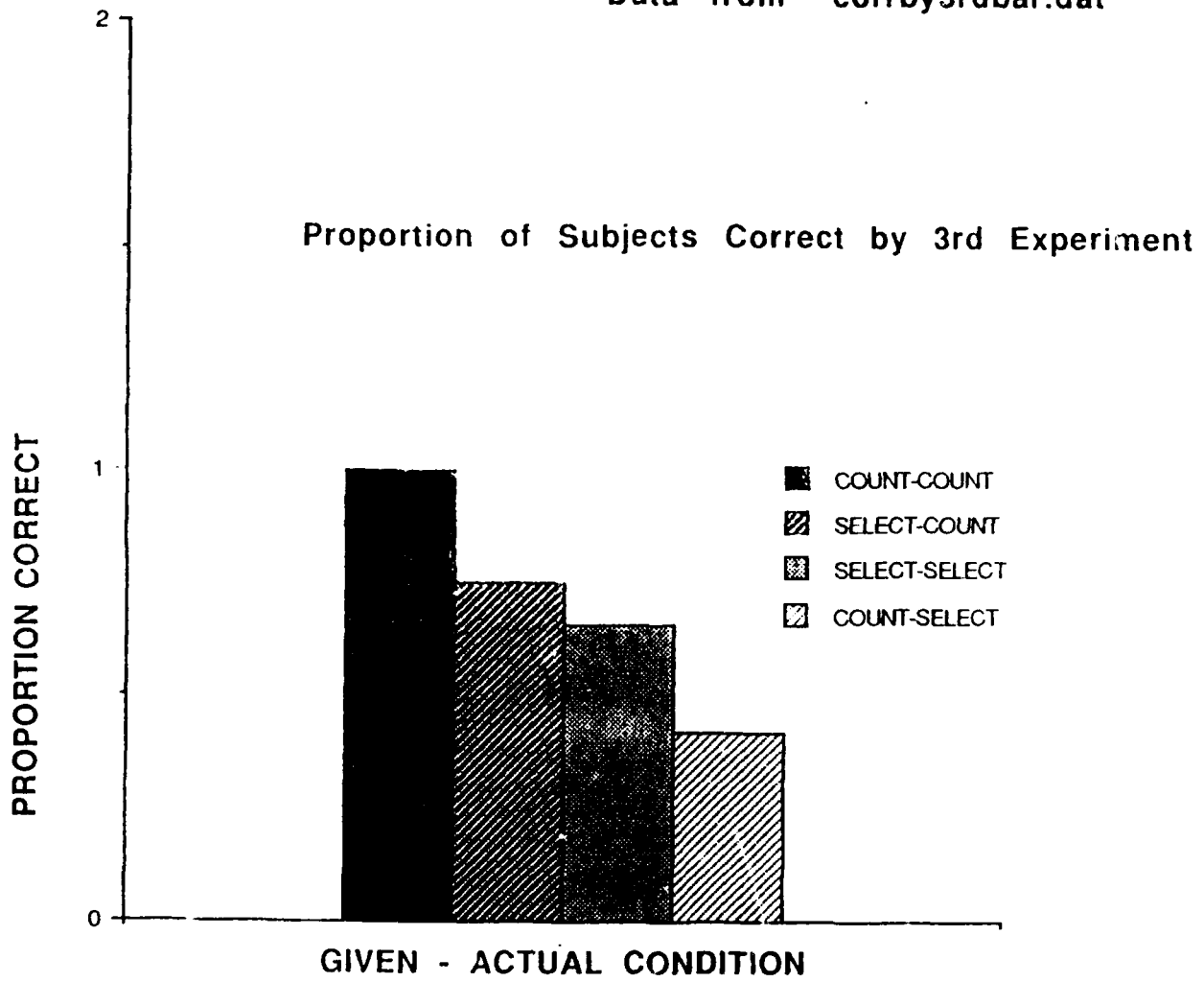
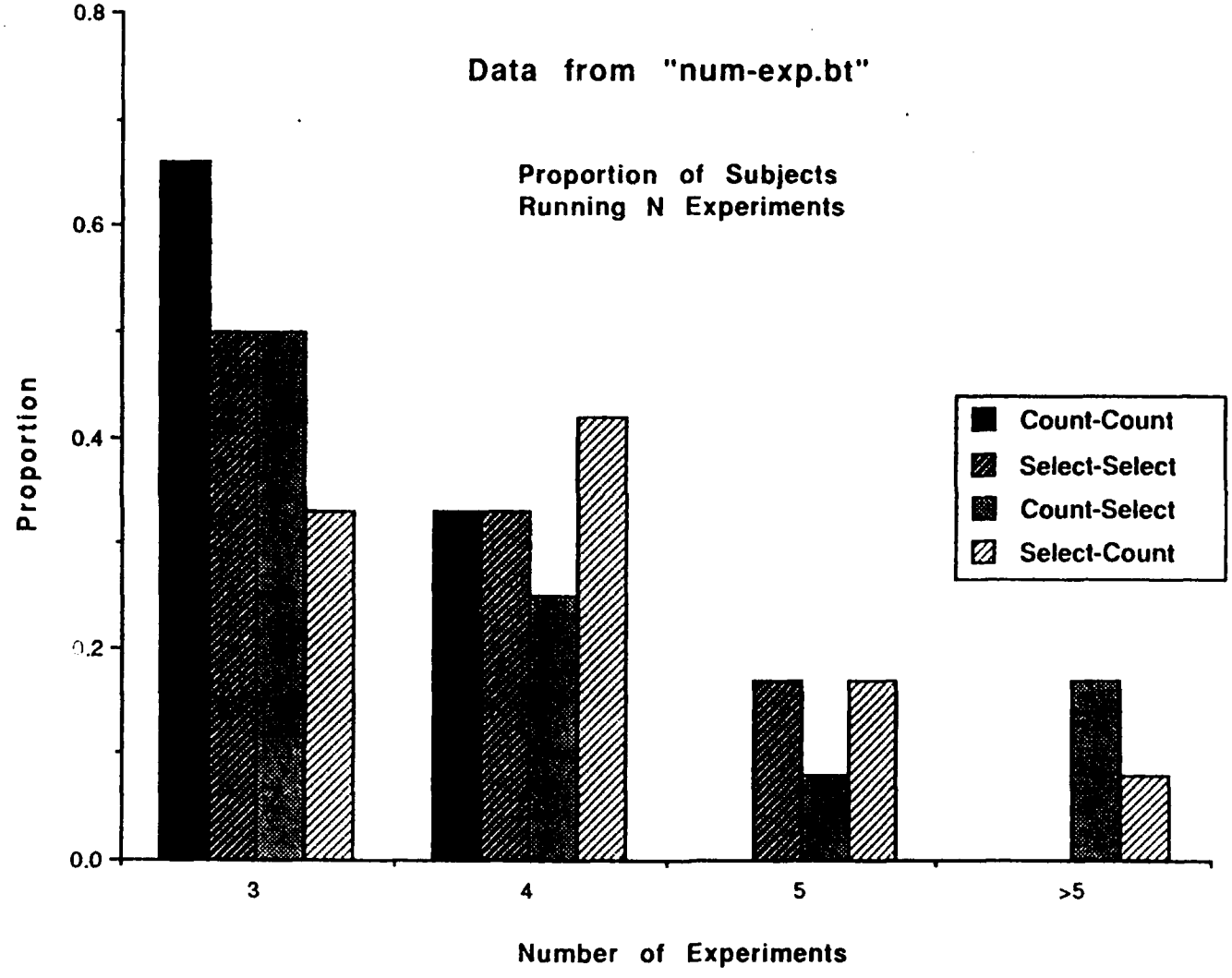

Data from "corrby3rdbar.dat"
Proportion of Subjects Correct by 3rd Experiment
PROPORTION CORRECT
2
1
0
COUNT-COUNT
SELECT-COUNT
SELECT-SELECT
COUNT-SELECT
GIVEN - ACTUAL CONDITION

Fig 5


Data from "num-exp.bt"
Proportion of Subjects
Running N Experiments
Proportion
0.8
0.6
0.4
0.2
0.0
Count-Count
Select-Select
Count-Select
Select-Count
3
4
5
>5
Number of Experiments